Docket No.: 57454-116 **PATENT**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of :

Toyohiko YOSHIDA, et al. :

Serial No.: : Group Art Unit:

Filed: May 16, 2001 : Examiner:

For: DATA PROCESSING APPARATUS OF HIGH SPEED PROCESS USING MEMORY OF LOW SPEED AND LOW POWER CONSUMPTION

CLAIM OF PRIORITY AND TRANSMITTAL OF CERTIFIED PRIORITY DOCUMENT

Commissioner for Patents
Washington, DC 20231

Sir:

In accordance with the provisions of 35 U.S.C. 119, Applicants hereby claim the priority of:

Japanese Patent Application No. 2000-257231,
filed August 28, 2000

cited in the Declaration of the present application. A certified copy is submitted herewith.

Respectfully submitted,

MCDERMOTT, WILL & EMERY

Stephen A. Becker
Registration No. 26,527

600 13th Street, N.W.
Washington, DC 20005-3096
(202) 756-8000 SAB:dtb
Date: May 16, 2001
Facsimile: (202) 756-8087

57454-116
Yoshida, et al
May 16, 2001
McDermott, Will & Emery

日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年　８月２８日

出願番号
Application Number: 特願２０００－２５７２３１

出願人
Applicant (s): 三菱電機株式会社

２０００年　９月１８日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２０００－３０７５１１７

【書類名】　特許願
【整理番号】　525067JP01
【提出日】　平成12年 8月28日
【あて先】　特許庁長官殿
【国際特許分類】　G11C 7/00
G06F 12/02
【発明者】
【住所又は居所】　東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
【氏名】　吉田　豊彦
【発明者】
【住所又は居所】　東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
【氏名】　山田　朗
【発明者】
【住所又は居所】　東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
【氏名】　佐藤　尚和
【特許出願人】
【識別番号】　000006013
【氏名又は名称】　三菱電機株式会社
【代理人】
【識別番号】　100064746
【弁理士】
【氏名又は名称】　深見　久郎
【選任した代理人】
【識別番号】　100085132
【弁理士】
【氏名又は名称】　森田　俊雄

【選任した代理人】

【識別番号】　100091409

【弁理士】

【氏名又は名称】　伊藤　英彦

【選任した代理人】

【識別番号】　100096781

【弁理士】

【氏名又は名称】　堀井　豊

【選任した代理人】

【識別番号】　100096792

【弁理士】

【氏名又は名称】　森下　八郎

【手数料の表示】

【予納台帳番号】　008693

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　データ処理装置

【特許請求の範囲】

【請求項１】　命令が格納される命令メモリと、

データが格納されるデータメモリと、

フェッチされた命令をデコードする命令デコーダと、

前記命令メモリ、前記データメモリおよび前記命令デコーダに接続され、前記命令メモリに格納される命令をフェッチし、前記命令デコーダのデコード結果に基づいて前記データメモリにアクセスするメモリ演算部と、

前記命令デコーダのデコード結果に基づいて整数演算を行なう整数演算部とを含むデータ処理装置であって、

前記命令メモリは、複数の命令メモリバンクを含み、

前記メモリ演算部は、前記複数の命令メモリバンクから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させてパイプライン処理を行なう、データ処理装置。

【請求項２】　前記命令メモリはさらに、連続したアドレスにアクセスした場合に、前記複数の命令メモリバンクの異なる命令メモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして前記複数の命令メモリバンクのチップセレクト信号を生成する第１のバンク選択回路を含む、請求項１記載のデータ処理装置。

【請求項３】　前記命令メモリはさらに、高速命令メモリを含み、

前記メモリ演算部は、前記高速命令メモリから命令をフェッチする場合に、前記命令メモリバンクの選択に対応したパイプラインステージを発生させずに、命令の読出しに対応したパイプラインステージを発生させてパイプライン処理を行なう、請求項１または２記載のデータ処理装置。

【請求項４】　前記データメモリは、複数のデータメモリバンクを含み、

前記メモリ演算部は、前記複数のデータメモリバンクにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージと、データのアクセ

スに対応したパイプラインステージとを発生させてパイプライン処理を行なう、請求項１～３のいずれかに記載のデータ処理装置。

【請求項５】　前記データメモリはさらに、前記複数のデータメモリバンクを２つの異なる領域に分割するために、上位アドレスを含んだアドレスをデコードして前記複数のデータメモリバンクのチップセレクト信号を生成する第２のバンク選択回路を含む、請求項４記載のデータ処理装置。

【請求項６】　前記第２のバンク選択回路は、前記２つの異なる領域の中の連続したアドレスにアクセスした場合に、前記複数のデータメモリバンクの異なるデータメモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして前記複数のデータメモリバンクのチップセレクト信号を生成する、請求項５記載のデータ処理装置。

【請求項７】　前記データメモリはさらに、高速データメモリを含み、

前記メモリ演算部は、前記高速データメモリにアクセスする場合に、前記データメモリバンクの選択に対応したパイプラインステージを発生させずに、データのアクセスに対応したパイプラインステージを発生させてパイプライン処理を行なう、請求項４～６のいずれかに記載のデータ処理装置。

【請求項８】　前記メモリ演算部は、命令バスを介して前記命令メモリからの命令のフェッチを行ない、前記命令バスと異なるデータバスを介して前記データメモリにアクセスする、請求項１～７のいずれかに記載のデータ処理装置。

【請求項９】　前記メモリ演算部は、データ入力バスを介して前記データメモリからデータを読出し、前記データ入力バスと異なるデータ出力バスを介して前記データメモリにデータを書込む、請求項１～８のいずれかに記載のデータ処理装置。

【請求項１０】　命令が格納される命令メモリと、

データが格納されるデータメモリと、

フェッチされた命令をデコードする命令デコーダと、

複数のレジスタを有するレジスタファイルと、

前記命令メモリ、前記データメモリおよび前記命令デコーダに接続され、前記命令メモリに格納される命令をフェッチし、前記命令デコーダのデコード結果に

基づいて前記データメモリにアクセスするメモリ演算部と、

前記命令デコーダのデコード結果に基づいて整数演算を行なう整数演算部とを含むデータ処理装置であって、

前記メモリ演算部は、リピート命令を実行する場合に、前記リピート命令の直後の命令を前記レジスタファイル内の専用レジスタに保持し、該専用レジスタに保持された命令をフェッチしながらリピート命令を実行する、データ処理装置。

【請求項１１】　前記レジスタファイルは、プロセッサ状態ワードを含み、

前記メモリ演算部は、リピート命令を実行する場合に、最初のループにおいて前記プロセッサ状態ワード内のフラグをセットし、前記命令メモリからフェッチした前記リピート命令の直後の命令を前記専用レジスタに保持し、

２回目のループにおいて前記プロセッサ状態ワード内の前記フラグをリセットし、前記専用レジスタに保持された命令をフェッチしながらリピート命令を実行する、請求項１０記載のデータ処理装置。

【請求項１２】　前記メモリ演算部は、リピート命令を実行する場合に、前記リピート命令の直後の複数の命令を前記レジスタファイル内の複数の専用レジスタに保持し、該複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行する、請求項１０記載のデータ処理装置。

【請求項１３】　前記レジスタファイルは、プロセッサ状態ワードを含み、

前記メモリ演算部は、リピート命令を実行する場合に、最初のループにおいて前記プロセッサ状態ワード内のフラグをセットし、前記命令メモリからフェッチした前記リピート命令の直後の複数の命令を前記複数の専用レジスタに保持し、

２回目のループにおいて前記プロセッサ状態ワード内の前記フラグをリセットし、前記複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行する、請求項１２記載のデータ処理装置。

【請求項１４】　前記命令メモリは、複数の命令メモリバンクを含み、

前記メモリ演算部は、前記複数の命令メモリバンクから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させてパイプライン処理を行なう、請求項１０～１３のいずれかに記載のデータ処理装置。

【請求項１５】　前記命令メモリはさらに、連続したアドレスにアクセスした場合に、前記複数の命令メモリバンクの異なる命令メモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして前記複数の命令メモリバンクのチップセレクト信号を生成する第１のバンク選択回路を含む、請求項１４記載のデータ処理装置。

【請求項１６】　前記データメモリは、複数のデータメモリバンクを含み、前記メモリ演算部は、前記複数のデータメモリバンクにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させてパイプライン処理を行なう、請求項１０～１５のいずれかに記載のデータ処理装置。

【請求項１７】　前記データメモリはさらに、前記複数のデータメモリバンクを２つの異なる領域に分割するために、上位アドレスを含んだアドレスをデコードして前記複数のデータメモリバンクのチップセレクト信号を生成する第２のバンク選択回路を含む、請求項１６記載のデータ処理装置。

【請求項１８】　前記第２のバンク選択回路は、前記２つの異なる領域の中の連続したアドレスにアクセスした場合に、前記複数のデータメモリバンクの異なるデータメモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして前記複数のデータメモリバンクのチップセレクト信号を生成する、請求項１７記載のデータ処理装置。

【請求項１９】　前記メモリ演算部は、タスクスイッチの切替え時に、前記専用レジスタを含んだ複数のレジスタを退避してタスクスイッチを切替える、請求項１０～１８のいずれかに記載のデータ処理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、内部メモリに記憶された命令およびデータをアクセスしながら処理を行なうデータ処理装置に関し、特に、低速で低消費電力のメモリを用いて高速に処理することが可能なデータ処理装置に関する。

【０００２】

【従来の技術】

近年、ＣＰＵ（Central Processing Unit）等のデータ処理装置が広く使用されており、データ処理装置の処理速度の向上に対する要望が高まる一方である。従来のデータ処理装置、たとえばＣＰＵにおいては、命令フェッチ機構、命令デコード機構および命令実行機構などがパイプライン化されている。そして、動作クロックの周波数を高くするとともにメモリのアクセスサイクルを短くし、１パイプラインステージタイム（動作クロックの１クロック）以内にメモリアクセスを行なうことによって、データ処理装置の処理性能を高めている。

【０００３】

しかし、アクセス時間の短い大容量メモリを実現するのは困難である。したがって、キャッシュメモリに代表されるように高速で小容量のメモリと、低速で大容量の主メモリとを階層的に構築し、見かけ上高速で大容量のメモリがあるかのように動作させてこの問題を解決していた。この階層化したメモリを用いた従来のデータ処理装置として、たとえば、J.L.Hennessy and D.A.Patterson, “Memory-Hierarchy Design,” Computer Architecture a Quantitative Approach,pp.372-483, Morgan kaufmann Publishers, Inc. San Francisco,1996.に詳しく紹介されている。

【０００４】

【発明が解決しようとする課題】

しかし、メモリを階層化することで擬似的に大容量の高速メモリを実現する場合、データ処理装置が小容量の高速メモリに入りきらない命令やデータをアクセスするとき（キャッシュミス）、メモリアクセスサイクルにウェイトサイクルを挿入する必要があり、データ処理装置の性能が低下するという問題点があった。また、メモリを高速動作させるためには、メモリに用いられるトランジスタの駆動能力を大きくする必要があるため、データ処理装置内に大容量の高速メモリを搭載するとメモリの消費電力が増大するという問題点があった。

【０００５】

本発明は、上記問題点を解決するためになされたものであり、第１の目的は、低速で低消費電力のメモリを用いつつ高スループットで処理することができ、処

理性能の向上を図ることが可能なデータ処理装置を提供することである。

【０００６】

第２の目的は、複数のメモリバンクに分割された命令メモリから命令をフェッチする構成において、リピート命令を実行した場合であっても同じメモリバンクに連続したアクセスが発生することを防止し、処理性能の向上を図ることが可能なデータ処理装置を提供することである。

【０００７】

第３の目的は、複数のメモリバンクに分割されたデータメモリにアクセスする構成において、変数データと係数データとを交互に読出す場合などにおいて同じメモリバンクに連続したアクセスが発生することを防止し、処理性能の向上を図ることが可能なデータ処理装置を提供することである。

【０００８】

【課題を解決するための手段】

請求項１に記載のデータ処理装置は、命令が格納される命令メモリと、データが格納されるデータメモリと、フェッチされた命令をデコードする命令デコーダと、命令メモリ、データメモリおよび命令デコーダに接続され、命令メモリに格納される命令をフェッチし、命令デコーダのデコード結果に基づいてデータメモリにアクセスするメモリ演算部と、命令デコーダのデコード結果に基づいて整数演算を行なう整数演算部とを含むデータ処理装置であって、命令メモリは、複数の命令メモリバンクを含み、メモリ演算部は、複数の命令メモリバンクから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させてパイプライン処理を行なう。

【０００９】

メモリ演算部は、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させるので、選択した命令メモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となる。また、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとが並列に行なわれるため、

命令メモリアクセスのスループットを向上させることが可能となる。

【００１０】

請求項２に記載のデータ処理装置は、請求項１記載のデータ処理装置であって、命令メモリはさらに、連続したアドレスにアクセスした場合に、複数の命令メモリバンクの異なる命令メモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして複数の命令メモリバンクのチップセレクト信号を生成する第１のバンク選択回路を含む。

【００１１】

したがって、同じ命令メモリバンクに連続してアクセスが行なわれなくなり、パイプラインの乱れを防止することが可能となる。

【００１２】

請求項３に記載のデータ処理装置は、請求項１または２記載のデータ処理装置であって、命令メモリはさらに、高速命令メモリを含み、メモリ演算部は、高速命令メモリから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージを発生させずに、命令の読出しに対応したパイプラインステージを発生させてパイプライン処理を行なう。

【００１３】

メモリ演算部は、高速命令メモリから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージを発生させないので、高速命令メモリアクセスのレイテンシを向上させることが可能となる。

【００１４】

請求項４に記載のデータ処理装置は、請求項１～３のいずれかに記載のデータ処理装置であって、データメモリは、複数のデータメモリバンクを含み、メモリ演算部は、複数のデータメモリバンクにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させてパイプライン処理を行なう。

【００１５】

メモリ演算部は、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させるので、選

択したデータメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となる。また、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとが並列に行なわれるため、データメモリアクセスのスループットを向上させることが可能となる。

【００１６】

請求項５に記載のデータ処理装置は、請求項４記載のデータ処理装置であって、データメモリはさらに、複数のデータメモリバンクを２つの異なる領域に分割するために、上位アドレスを含んだアドレスをデコードして複数のデータメモリバンクのチップセレクト信号を生成する第２のバンク選択回路を含む。

【００１７】

したがって、変数データと係数データとを交互に読出す場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００１８】

請求項６に記載のデータ処理装置は、請求項５記載のデータ処理装置であって、第２のバンク選択回路は、２つの異なる領域の中の連続したアドレスにアクセスした場合に、複数のデータメモリバンクの異なるデータメモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして複数のデータメモリバンクのチップセレクト信号を生成する。

【００１９】

したがって、ポストインクリメントを用いてデータメモリにアクセスする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００２０】

請求項７に記載のデータ処理装置は、請求項４～６のいずれかに記載のデータ処理装置であって、データメモリはさらに、高速データメモリを含み、メモリ演算部は、高速データメモリにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージを発生させずに、データのアクセスに対応したパ

イプラインステージを発生させてパイプライン処理を行なう。

【００２１】

メモリ演算部は、高速データメモリにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージを発生させないので、高速データメモリアクセスのレイテンシを向上させることが可能となる。

【００２２】

請求項８に記載のデータ処理装置は、請求項１～７のいずれかに記載のデータ処理装置であって、メモリ演算部は、命令バスを介して命令メモリからの命令のフェッチを行ない、命令バスと異なるデータバスを介してデータメモリにアクセスする。

【００２３】

したがって、命令メモリからの命令のフェッチと、データメモリへのアクセスとのバスの競合が発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００２４】

請求項９に記載のデータ処理装置は、請求項１～８のいずれかに記載のデータ処理装置であって、メモリ演算部は、データ入力バスを介してデータメモリからデータを読出し、データ入力バスと異なるデータ出力バスを介してデータメモリにデータを書込む。

【００２５】

したがって、データの読出しと、データの書込みとのバスの競合が発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００２６】

請求項１０に記載のデータ処理装置は、命令が格納される命令メモリと、データが格納されるデータメモリと、フェッチされた命令をデコードする命令デコーダと、複数のレジスタを有するレジスタファイルと、命令メモリ、データメモリおよび命令デコーダに接続され、命令メモリに格納される命令をフェッチし、命令デコーダのデコード結果に基づいてデータメモリにアクセスするメモリ演算部と、命令デコーダのデコード結果に基づいて整数演算を行なう整数演算部とを含

むデータ処理装置であって、メモリ演算部は、リピート命令を実行する場合に、リピート命令の直後の命令をレジスタファイル内の専用レジスタに保持し、専用レジスタに保持された命令をフェッチしながらリピート命令を実行する。

【００２７】

メモリ演算部は、専用レジスタに保持された命令をフェッチしながらリピート命令を実行するので、複数のメモリバンクに分割された命令メモリから命令をフェッチする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、処理性能の向上を図ることが可能となる。

【００２８】

請求項１１に記載のデータ処理装置は、請求項１０記載のデータ処理装置であって、レジスタファイルは、プロセッサ状態ワードを含み、メモリ演算部は、リピート命令を実行する場合に、最初のループにおいてプロセッサ状態ワード内のフラグをセットし、命令メモリからフェッチしたリピート命令の直後の命令を専用レジスタに保持し、２回目のループにおいてプロセッサ状態ワード内のフラグをリセットし、専用レジスタに保持された命令をフェッチしながらリピート命令を実行する。

【００２９】

したがって、プロセッサ状態ワードのフラグを参照することにより、リピート命令の実行状況を制御することが可能となる。

【００３０】

請求項１２に記載のデータ処理装置は、請求項１０記載のデータ処理装置であって、メモリ演算部は、リピート命令を実行する場合に、リピート命令の直後の複数の命令をレジスタファイル内の複数の専用レジスタに保持し、複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行する。

【００３１】

メモリ演算部は、複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行するので、複数のメモリバンクに分割された命令メモリから命令をフェッチする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、処理性能の向上を図ることが可能となる。

【００３２】

請求項１３に記載のデータ処理装置は、請求項１２記載のデータ処理装置であって、レジスタファイルは、プロセッサ状態ワードを含み、メモリ演算部は、リピート命令を実行する場合に、最初のループにおいてプロセッサ状態ワード内のフラグをセットし、命令メモリからフェッチしたリピート命令の直後の複数の命令を複数の専用レジスタに保持し、２回目のループにおいてプロセッサ状態ワード内のフラグをリセットし、複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行する。

【００３３】

したがって、プロセッサ状態ワードのフラグにより、リピート命令の実行状況を制御することが可能となる。

【００３４】

請求項１４に記載のデータ処理装置は、請求項１０～１３のいずれかに記載のデータ処理装置であって、命令メモリは、複数の命令メモリバンクを含み、メモリ演算部は、複数の命令メモリバンクから命令をフェッチする場合に、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させてパイプライン処理を行なう。

【００３５】

したがって、選択した命令メモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となる。また、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとが並列に行なわれるため、命令メモリアクセスのスループットを向上させることが可能となる。

【００３６】

請求項１５に記載のデータ処理装置は、請求項１４記載のデータ処理装置であって、命令メモリはさらに、連続したアドレスにアクセスした場合に、複数の命令メモリバンクの異なる命令メモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして複数の命令メモリバンクのチップセレクト信号を生成する第１のバンク選択回路を含む。

【００３７】

したがって、同じ命令メモリバンクに連続してアクセスが行なわれなくなり、パイプラインの乱れを防止することが可能となる。

【００３８】

請求項１６に記載のデータ処理装置は、請求項１０～１５のいずれかに記載のデータ処理装置であって、データメモリは、複数のデータメモリバンクを含み、メモリ演算部は、複数のデータメモリバンクにアクセスする場合に、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させてパイプライン処理を行なう。

【００３９】

したがって、選択したデータメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となる。また、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとが並列に行なわれるため、データメモリアクセスのスループットを向上させることが可能となる。

【００４０】

請求項１７に記載のデータ処理装置は、請求項１６記載のデータ処理装置であって、データメモリはさらに、複数のデータメモリバンクを２つの異なる領域に分割するために、上位アドレスを含んだアドレスをデコードして複数のデータメモリバンクのチップセレクト信号を生成する第２のバンク選択回路を含む。

【００４１】

したがって、変数データと係数データとを交互に読出す場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００４２】

請求項１８に記載のデータ処理装置は、請求項１７記載のデータ処理装置であって、第２のバンク選択回路は、２つの異なる領域の中の連続したアドレスにアクセスした場合に、複数のデータメモリバンクの異なるデータメモリバンクにアクセスされるように、下位アドレスを含んだアドレスをデコードして複数のデー

タメモリバンクのチップセレクト信号を生成する。

【００４３】

したがって、ポストインクリメントを用いてデータメモリにアクセスする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となる。

【００４４】

請求項１９に記載のデータ処理装置は、請求項１０～１８のいずれかに記載のデータ処理装置であって、メモリ演算部は、タスクスイッチの切替え時に、専用レジスタを含んだ複数のレジスタを退避してタスクスイッチを切替える。

【００４５】

したがって、元のタスクに復帰する際に、容易にタスクスイッチを切替えることが可能となる。

【００４６】

【発明の実施の形態】

図１は、本発明の実施の形態におけるプロセッサを用いたデータ処理装置の概略構成を示すブロック図である。このデータ処理装置は、プロセッサ１０と、バス制御回路２０と、ＲＯＭ（Read Only Memory）２１と、ＳＤＲＡＭ（Synchronous Dynamic Random Access Memory）２２とを含む。また、プロセッサ１０、バス制御回路２０、ＲＯＭ２１およびＳＤＲＡＭ２２は、アドレスバス３１、データバス３２および制御バス３３によって接続されている。

【００４７】

プロセッサ１０は、アドレスバス３１を介してバス制御回路２０、ＲＯＭ２１およびＳＤＲＡＭ２２へアドレスを出力する。プロセッサ１０がＳＤＲＡＭ２２にデータを書込む場合には、データバス３２を介してＳＤＲＡＭ２２へデータを出力する。プロセッサ１０がＲＯＭ２１またはＳＤＲＡＭ２２からデータを読出す場合には、ＲＯＭ２１またはＳＤＲＡＭ２２から出力されたデータをデータバス３２を介して入力する。また、バス制御回路２０は、プロセッサ１０から出力される制御信号を受けて、ＲＯＭ２１やＳＤＲＡＭ２２を制御するための信号を生成して出力する。

【００４８】

図２は、本実施の形態におけるプロセッサ１０の概略構成を示すブロック図である。このプロセッサ１０は、ＶＬＩＷ(Very Long Instruction Word) 方式のＣＰＵコア（以下、単にコアと呼ぶ。）１００と、高速命令メモリ１０１と、高速データメモリ１０２と、低電力命令メモリ１０３と、低電力データメモリ１０４と、アドレスバス３１、データバス３２および制御バス３３を介して外部のバス制御回路２０、ＲＯＭ２１およびＳＤＲＡＭ２２に接続されるシステムバスＩ／Ｆ（以下、バスインタフェース部と呼ぶ。）１０５とを含む。

【００４９】

コア１００、高速命令メモリ１０１、高速データメモリ１０２、低電力命令メモリ１０３および低電力データメモリ１０４は、データアドレスバス１０６、データ出力バス１０７およびデータ入力バス１０８に接続されている。また、データと命令とを並列にアクセスできるように、コア１００、高速命令メモリ１０１、低電力命令メモリ１０３およびバスインタフェース部１０５は、命令アドレスバス１０９および命令バス１１０にも接続されている。

【００５０】

また、コア１００は、２ウェイのＶＬＩＷ型命令体系を有しており、２つのサブ命令を含んだＶＬＩＷ命令を実行する。コア１００は、命令バス１１０を介して入力されたＶＬＩＷ命令をデコードする命令デコーダ１１３と、レジスタファイル１２０と、アドレス演算命令を実行するメモリ演算部１３０と、整数演算命令を実行する整数演算部１４０とを含む。

【００５１】

命令デコーダ１１３は、ＶＬＩＷ命令に含まれるそれぞれのサブ命令をデコードするサブ命令デコーダ１１１および１１２を含む。メモリ演算部１３０は、メモリアドレス演算器１３１と、ＰＣ（Program Counter）演算器１３２、シフタ１３３と、ＡＬＵ（Arithmetic and Logic Unit）１３４とを含む。また、整数演算部１４０は、シフタ１４１と、ＡＬＵ１４２と、乗算器１４３と、６４ビットのアキュムレータ１４４とを含む。

【００５２】

メモリ演算部１３０および整数演算部１４０は、２つのサブ命令デコーダ１１１および１１２のデコード結果に従ってそれぞれのサブ命令を実行するが、２つのサブ命令を並列に実行する場合と、２つのサブ命令を順番に実行する場合とがある。レジスタファイル１２０は、ソフトウェアによって読み書き可能な６４本の汎用レジスタを含む。

【００５３】

図３は、コア１００が有するレジスタを説明するための図である。汎用レジスタＲ０～Ｒ６３は、上述したレジスタファイル１２０内に存在する。レジスタＲ０は、常に“０”を保持するレジスタであり、他のレジスタのクリア等に使用される。レジスタＲ６２は、サブルーチンの戻り先アドレスを保持するためのリンクポインタである。レジスタＲ６３は、非割込み処理中におけるスタックポインタおよび割込み処理中におけるスタックポインタであり、後述するＰＳＷ（Processor Status Word）内のモードビットＳＭによって切替えられる。

【００５４】

アキュムレータＡ０およびＡ１は、乗算結果または積和演算結果を保持するためのレジスタであり、汎用レジスタの２倍のビット長である６４ビットのレジスタである。

【００５５】

レジスタＲＰＴ０＿Ｃ、ＲＰＴ０＿Ｓ、ＲＰＴ０＿ＥおよびＲＰＴ０＿Ｉは、リピート命令ＲＥＰＥＡＴ０に従ってハードウェアループ制御が行なわれるときに使用されるレジスタである。ＲＰＴ０＿Ｃは、ループカウンタ値を保持する。ＲＰＴ０＿ＳおよびＲＰＴ０＿Ｅは、それぞれループの先頭命令のアドレスおよび最後尾命令のアドレスを保持する。ＲＰＴ０＿Ｉは、ループの先頭命令の命令コードを保持する。

【００５６】

レジスタＲＰＴ１＿Ｃ、ＲＰＴ１＿Ｓ、ＲＰＴ１＿ＥおよびＲＰＴ１＿Ｉ（ｎ）は、リピート命令ＲＥＰＥＡＴ１に従ってハードウェアループ制御が行なわれるときに使用されるレジスタである。ＲＰＴ１＿Ｃは、ループカウンタ値を保持する。ＲＰＴ１＿ＳおよびＲＰＴ１＿Ｅは、それぞれループの先頭命令のアドレ

スおよび最後尾命令のアドレスを保持する。ＲＰＴ１＿Ｉ（ｎ）は、ループの先頭命令から順に６個の命令コードを保持する。

【００５７】

レジスタＰＳＷは、プロセッサ状態ワードであり、コア１００を制御するためのフラグ等を保持するレジスタである。レジスタＰＣは、コア１００が現在実行しているプログラムのアドレスを保持するレジスタである。レジスタＢＰＳＷおよびＢＰＣは、それぞれバックアップ用のＰＳＷおよびＰＣであり、割り込み等の事象が発生した場合にＰＳＷおよびＰＣの値が自動的にコピーされる。

【００５８】

レジスタＭＯＤ０＿Ｓ、ＭＯＤ０＿Ｅ、ＭＯＤ１＿ＳおよびＭＯＤ１＿Ｅは、ループバッファなどに使用するモジュロアドレッシングの制御を行なうためのレジスタである。レジスタＭＯＤ０＿ＳおよびＭＯＤ０＿Ｅがペアとなって、第１のループバッファの先頭アドレスおよび最後尾アドレスを保持する。また、レジスタＭＯＤ１＿ＳおよびＭＯＤ１＿Ｅがペアとなって、第２のループバッファの先頭アドレスおよび最後部アドレスを保持する。

【００５９】

レジスタＩＢＡは、デバッガがブレークポイントアドレスを指定する際にその値を保持するレジスタである。

【００６０】

図４は、ＰＳＷの詳細を説明するための図である。ＳＭは、割込み処理中か非割り込み処理中かを示すビットであり、上述したレジスタＲ６３（ＳＰＵ）とＲ６３（ＳＰＩ）とを切替えるためのモードビットである。ＤＢは、デバッグ中か否かを示すビットであり、“１”のときに上述したレジスタＩＢＡが有効となる。ＩＥは、割込み許可中か割込み禁止中かを示すビットであり、割込み許可中のときに外部から割込み要求があるとＶＬＩＷ命令の切れ目で割込みを受付ける。

【００６１】

ＲＰ０およびＲＰ１は、それぞれリピート命令ＲＥＰＥＡＴ０またはＲＥＰＥＡＴ１の実行によってハードウェアループ制御がイネーブルになると、“１”になるビットである。また、ＦＳ０およびＦＳ１は、それぞれリピート命令ＲＥＰ

ＥＡＴ０またはＲＥＰＥＡＴ１の実行において、１回目のハードウェアループのときにのみ“１”となるビットである。

【００６２】

ＭＤ０およびＭＤ１は、それぞれＭＯＤ０＿ＳおよびＭＯＤ０＿Ｅと、ＭＯＤ１＿ＳおよびＭＯＤ１＿Ｅとによるモジュロアドレッシングをイネーブルにするか、ディスエイブルにするかを決定するビットである。Ｆ０～Ｆ７は、命令の実行条件を制御するためのビットである。

【００６３】

図５は、低電力命令メモリ１０３の概略構成を示すブロック図である。この低電力命令メモリ１０３は、８個のメモリバンク４０～４７と、メモリバンク４０～４７のいずれかを選択するバンク選択回路４８とを含む。バンク選択回路４８は、データアドレスバス１０６、データ出力バス１０７、命令アドレスバス１０９および命令バス１１０に接続される。また、バンク選択回路４８は、コア１００から出力されるＢＳＩ（命令を示すバスステータス）信号およびＢＳＤ（データを示すバスステータス）信号を受けて、メモリバンク４０～４７のＣＳ（チップセレクト）信号を生成する。

【００６４】

低電力命令メモリ１０３は、スループットが１クロックサイクル、レイテンシが２クロックサイクルのメモリである。この理由については後述する。また、コア１００からのアドレス転送と読出した命令コードの転送とにそれぞれ１／２クロックサイクルが使用されるので、コア１００による命令コードのフェッチにおけるレイテンシが３クロックサイクルとなる。なお、コア１００からのアドレス転送および読出した命令コードの転送がメモリアクセスとパイプライン化されているため、スループットは１クロックサイクルのままである。

【００６５】

異なるメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが１クロックサイクルとなってコア１００が高速に命令をフェッチすることが可能となる。また、同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなって１クロックサイクルの

無駄が発生する。

【００６６】

図６は、バンク選択回路４８の詳細を説明するためのブロック図である。このバンク選択回路４８は、データアドレス信号１０６が入力されるラッチ５６と、命令アドレス信号１０９が入力されるラッチ５７と、ラッチ５６および５７からの出力を選択するマルチプレクサ５５と、アドレス入力レジスタ５１と、データ入力レジスタ５２と、命令出力レジスタ５３と、メモリバンク４０～４７のＣＳ信号を生成するＣＳ信号生成回路５４とを含む。

【００６７】

マルチプレクサ５５は、ラッチ５６によって保持されたデータアドレス信号と、ラッチ５７によって保持された命令アドレス信号とを切替えて出力する。アドレス入力レジスタ５１は、マルチプレクサ５５から出力されたアドレスを保持する。データ入力レジスタ５２は、データ出力バス１０７の内容を保持する。命令出力レジスタ５３は、メモリバンク４０～４７から出力された命令コードを保持して命令バス１１０へ出力する。

【００６８】

アドレス入力レジスタ５１、データ入力レジスタ５２および命令出力レジスタ５３は、それぞれダブルバッファになっている。同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなって１クロックサイクルの無駄が発生する。そのため、２クロックサイクルの間値を保持しつつ、次の命令のアドレス等を保持するためにダブルバッファの構成を採用している。

【００６９】

ＣＳ信号生成回路５４は、ＢＳＩ信号、ＢＳＤ信号およびアドレス入力レジスタ５１に保持されるアドレス（Ａ０～Ａ１６、Ａ２７、Ａ２８）に基づいて、メモリバンク４０～４７のＣＳ信号を生成する。上述したように、同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなるため１ウェイト挿入される。ＷａｉｔＩ信号およびＷａｉｔＤ信号は、それぞれ命令またはデータのアクセスにおいてウェイトを挿入する際にア

クティブとなる。

【００７０】

図７は、メモリバンクのＣＳ信号の生成を説明するための図である。図７に示すように、３２ビットアドレスのうちＡ０～Ａ１５をデコードしてメモリバンク４０～４７へのアクセスであることを判別する。また、メモリバンク４０～４７のいずれのメモリバンクへのアクセスであるかを、Ａ１６、Ａ２７およびＡ２８の３ビットをデコードすることによって判別する。低電力命令メモリ１０３は８バイト単位でアクセスされるため、Ａ２９～Ａ３１の値は任意である。なお、図７において、“H’”は１６進数であることを示している。

【００７１】

図８は、メモリバンク４０～４７にアクセスするときのパイプライン処理を説明するためのタイミングチャートである。図８は、スループットが１クロックサイクルとなり、レイテンシが２クロックサイクルとなることを説明するためのものである。パイプライン処理の詳細な説明は後述する。

【００７２】

図８において、ＩＦ０～ＩＦ２、Ｄ、Ｒ、Ｍ０～Ｍ２およびＷは、パイプラインのステージを示している。命令フェッチステージＩＦ０において、最初の１／２クロックで命令アドレスバス１０９を介して命令アドレスが転送される。次の１／２クロックでメモリバンク４０～４７の選択が行なわれる。

【００７３】

命令フェッチステージＩＦ１において、選択されたメモリバンクのプリチャージが最初の１／２クロックで行なわれる。このタイミングでＣＳ信号生成回路５４からメモリバンクのＣＳ信号が出力されて、選択されたメモリバンク内のビット線が活性化される。そして、次の１／２クロックと命令フェッチステージＩＦ２の最初の１／２クロックとで命令のフェッチが行なわれる。なお、命令フェッチステージＩＦ１において、次の命令をフェッチするために次の命令のアドレスバス転送が行なわれ、パイプライン的に処理が行なわれる。

【００７４】

命令デコードステージＤにおいて、フェッチされた命令がデコードされる。フ

ェッチされた命令がロード命令である場合、リードレジスタステージＲにおいてレジスタからデータアドレスが読出される。次に、データメモリアクセスステージＭ０において、データアドレスバス転送およびバンクの選択が行なわれる。次に、データメモリアクセスステージＭ１およびＭ２において、データの読出しおよびデータバス転送が行なわれる。そして、ライトバックステージＷにおいて、読出したデータがレジスタに書込まれる。なお、低電力データメモリ１０４に対するアクセスのタイミングは、低電力命令メモリ１０３に対するアクセスのタイミングと同じである。

【００７５】

通常、コア１００が命令をフェッチする場合には、連続したアドレスにアクセスするため、アドレスのＡ２７およびＡ２８が「００」、「０１」、「１０」、「１１」とサイクリックに変化する。したがって、必ず異なるメモリバンクにアクセスすることになり、同じメモリバンクに続けてアクセスすることはない。

【００７６】

図９は、低電力データメモリ１０４の概略構成を示すブロック図である。この低電力データメモリ１０４は、８個のメモリバンク６０～６７と、メモリバンク６０～６７のいずれかを選択するバンク選択回路６８とを含む。バンク選択回路６８は、データアドレスバス１０６、データ出力バス１０７およびデータ入力バス１０８に接続される。また、バンク選択回路６８は、コア１００から出力されるＢＳＤ信号を受けて、メモリバンク６０～６７のＣＳ信号を生成する。

【００７７】

低電力データメモリ１０４は、スループットが１クロックサイクル、レイテンシが２クロックサイクルのメモリである。また、コア１００からのアドレス転送と読出したデータの転送とにそれぞれ１／２クロックサイクルが使用されるので、コア１００によるメモリアクセスにおけるレイテンシが３クロックサイクルとなる。なお、コア１００からのアドレス転送および読出したデータの転送がメモリアクセスとパイプライン化されているため、スループットは１クロックサイクルのままである。

【００７８】

異なるメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが１クロックサイクルとなってコア１００が高速にメモリアクセスを行なうことが可能となる。また、同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなって１クロックサイクルの無駄が発生する。

【００７９】

図１０は、バンク選択回路６８の詳細を説明するためのブロック図である。このバンク選択回路６８は、アドレス入力レジスタ７１と、データ入力レジスタ７２と、データ出力レジスタ７３と、メモリバンク６０～６７のＣＳ信号を生成するＣＳ信号生成回路７４とを含む。

【００８０】

アドレス入力レジスタ７１、データ入力レジスタ７２およびデータ出力レジスタ７３は、それぞれダブルバッファになっている。同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなって１クロックサイクルの無駄が発生する。そのため、２クロックサイクルの間値を保持しつつ、次のデータのアドレス等を保持するためにダブルバッファの構成を採用している。

【００８１】

ＣＳ信号生成回路７４は、ＢＳＤ信号およびアドレス入力レジスタ７１に保持されるアドレス（Ａ０～Ａ１６、Ａ２７、Ａ２８）に基づいて、メモリバンク６０～６７のＣＳ信号を生成する。上述したように、同じメモリバンクに対して連続してアクセスが行なわれた場合には、スループットが２クロックサイクルとなるため１ウェイト挿入される。ＷａｉｔＤ信号は、メモリアクセスにおいてウェイトを挿入する際にアクティブとなる。

【００８２】

図１１は、メモリバンクのＣＳ信号の生成を説明するための図である。図１１に示すように、３２ビットアドレスのうちＡ０～Ａ１５をデコードしてメモリバンク６０～６７へのアクセスであることを判別する。また、メモリバンク６０～６７のいずれのメモリバンクへのアクセスであるかを、Ａ１６、Ａ２７およびＡ

２８の３ビットをデコードすることによって判別する。Ａ１６は、メモリバンク６０～６３へのアクセスであるか、メモリバンク６４～６７へのアクセスであるかを示している。低電力データメモリ１０４は８バイト単位でアクセスされるため、Ａ２９～Ａ３１の値は任意である。

【００８３】

デジタル信号処理においては、連続したアドレスのデータが順次アクセスされる場合が多い。８バイト単位でデータがアクセスされる場合には、アドレスのＡ２７およびＡ２８が「００」、「０１」、「１０」、「１１」とサイクリックに変化する。したがって、必ず異なるメモリバンクにアクセスすることになり、同じメモリバンクに続けてアクセスすることはない。

【００８４】

また、１、２、４バイト単位でデータがアクセスされる場合には、最初のメモリアクセスで一旦８バイトのデータがデータ出力レジスタ７３に格納される。そして、２回目以降のメモリアクセスにおいては、メモリバンクへのアクセスは行なわれずに、データ出力レジスタ７３に格納されたデータが順次データ入力バス１０８に出力される。したがって、スループットが１に保たれる。

【００８５】

デジタル信号処理においては、低電力データメモリ１０４から同数の変数と係数とが読出される場合が多い。したがって、アドレスのＡ１６の値によって変数が格納される領域（バンクメモリ６０～６３）と係数が格納される領域（バンクメモリ６４～６７）とを分け、変数と係数とを交互に読出す場合でも同じバンクメモリに対するアクセスが発生しないようにしている。

【００８６】

図１２は、コア１００が実行する命令のフォーマットを説明するための図である。コア１００が実行する命令は、２ウェイのＶＬＩＷ型命令であり、サブ命令を格納するＬコンテナ２０５およびＲコンテナ２０６と、各サブ命令の実行条件を指定するＣＣフィールド２０３および２０４と、各サブ命令の実行順序または長いサブ命令を定義するＦＭフィールド２０１ａおよび２０１ｂとを含む。

【００８７】

ＣＣフィールド２０３および２０４は、ＰＳＷ中のフラグＦ０およびＦ１に依存した条件を指定する。たとえば、ＣＣフィールド２０３が“０００”のとき、Ｌコンテナ２０５に含まれるサブ命令が無条件に実行される。また、ＣＣフィールド２０４が“１０１”のとき、Ｒコンテナ２０６に含まれるサブ命令が、フラグＦ０＝“１”かつＦ１＝“１”の場合に実行され、フラグＦ０およびＦ１がそれ以外の値の場合にはサブ命令が無効化される。

【００８８】

ＦＭフィールド２０１ａおよび２０１ｂは、Ｌコンテナ２０５およびＲコンテナ２０６に含まれるサブ命令の実行順序または長いサブ命令を定義する。ＦＭフィールドが“００”のとき、Ｌコンテナ２０５およびＲコンテナ２０６に含まれる２つのサブ命令が並列に実行される。ＦＭフィールドが“０１”のとき、Ｌコンテナ２０５に含まれるサブ命令がまず実行され、次にＲコンテナ２０６に含まれるサブ命令が実行される。ＦＭフィールドが“１０”のとき、Ｒコンテナ２０６に含まれるサブ命令がまず実行され、次にＬコンテナ２０５に含まれるサブ命令が実行される。ＦＭフィールドが“１１”のとき、Ｌコンテナ２０５およびＲコンテナ２０６に分割して保持された１つの長いサブ命令が実行される。

【００８９】

図１３（ａ）～図１３（ｈ）は、Ｌコンテナ２０５およびＲコンテナ２０６に保持されるサブ命令のフォーマットを示す図である。短いサブ命令は２８ビットの長さを有し、７種類のフォーマットに分類される。図１３（ａ）～図１３（ｇ）に示すように、短いサブ命令のビット位置０～９で演算の種類が指定され、ビット位置１０～２７で最大３つのオペランドが指定される。長いサブ命令は５４ビットの長さを有し、図１３（ｈ）に示すように、長いサブ命令のビット位置０～９で演算の種類が指定され、ビット位置１０～５３で３２ビット長の即値データを含む最大３つのオペランドが指定される。なお、長いサブ命令の３２ビットの即値は、図１２に示すＶＬＩＷ命令におけるビット位置２６～３１、３６～４３および４６～６３に保持される。

【００９０】

図１３（ａ）に示すフォーマットは、メモリアクセス演算（ＬＤ命令／ＳＴ命

令）を行なうサブ命令のフォーマットである。このサブ命令は、演算内容（オペコード）を指定するフィールド（ビット位置０～７）と、レジスタであるか即値であるかを指定するフィールドＸ（ビット位置８～９）、レジスタ番号を指定するフィールドＲａ（ビット位置１０～１５）およびＲｂ（ビット位置１６～２１）と、レジスタ番号または6ビット長の即値を指定するフィールドｓｒｃ（ビット位置２２～２７）とを含む。図１３（ａ）に示すように、フィールドＸの値が“００”、“０１”または“１１”である場合は、フィールドｓｒｃがレジスタ番号であることを示しており、フィールドＸの値が“１０”である場合は、フィールドｓｒｃが即値であることを示している。このサブ命令は、レジスタ間接アドレッシングによるメモリアクセス演算に用いられる。なお、“Ｒｂ＋＋”および“Ｒｂ－－”は、レジスタ間接アドレッシングの際のアドレッシングモードを示しており、“Ｒｂ＋＋”はポストインクリメント付きレジスタ間接モードを、“Ｒｂ－－”はポストデクリメント付きレジスタ間接モードを示している。

【００９１】

図１３（ｂ）に示すフォーマットは、汎用レジスタに保持されたオペランド間の演算（ＡＬＵ命令）または積和演算（ＭＡＣ命令）を行なうサブ命令のフォーマットである。このサブ命令は、レジスタ番号または即値を指定するフィールドＹ（ビット位置８）を含む。図１３（ｂ）に示すように、フィールドＹの値が“０”である場合は、ｓｒｃがレジスタ番号であることを示しており、フィールドＹの値が“１”である場合は、ｓｒｃが即値であることを示している。

【００９２】

図１３（ｃ）～図１３（ｇ）に示すフォーマットは、ブランチ演算（ＢＲＡ命令）を行なうサブ命令のフォーマットである。図１３（ｃ）～図１３（ｇ）に示すように、フィールドＦ（ビット位置８）によってレジスタであるか、即値であるかが指定され、フィールドＺ（ビット位置９）によってゼロフラグを参照した分岐命令であることが指定される。また、ビット位置１０～２７に示すフィールドによって、レジスタまたは即値による分岐先アドレスが指定される。

【００９３】

図１３（ｈ）に示すフォーマットは、長いサブ命令のフォーマットを示してい

る。このサブ命令は、演算内容（オペコード）を指定するフィールド（ビット位置０～７）と、レジスタ番号を指定するフィールドＲａ（ビット位置１０～１５）およびＲｂ（ビット位置１６～２１）と、３２ビット長の即値を指定するフィールドｉｍｍ（ビット位置２２～５３）とを含む。この長いサブ命令は、メモリアクセス演算、汎用レジスタに保持されたオペランド間の演算、積和演算およびブランチ演算のいずれにも使用される。

【００９４】

図１４は、本実施の形態におけるコア１００のパイプライン処理を説明するための図である。図１４（ａ）～図１４（ｄ）は、低電力命令メモリ１０３からフェッチした命令を実行するときのパイプライン処理を示しており、それぞれＡＬＵ命令、ＭＡＣ命令、ＬＤ／ＳＴ命令およびＢＲＡ命令を示している。ステージＩＦ０～ＩＦ２は、命令フェッチステージを示している。ステージＤは、命令デコードステージを示している。

【００９５】

ステージＲは、レジスタファイル１２０からオペランドを読出すステージである。ステージＲ／Ａは、レジスタファイル１２０からＰＣ値の読出し、または読出したＰＣ値にディスプレースメント値を加算して分岐先アドレスを計算するステージである。ステージＥ０およびＥ１は、命令実行のための演算を行なうステージである。ステージＭ０～Ｍ２は、データメモリに対してアクセスを行なうステージである。ステージＷは、オペランドを汎用レジスタに書込むステージである。

【００９６】

図１４（ｅ）～図１４（ｈ）は、高速命令メモリ１０１からフェッチした命令を実行するときのパイプライン処理を示しており、それぞれＡＬＵ命令、ＭＡＣ命令、ＬＤ／ＳＴ命令およびＢＲＡ命令を示している。これらのパイプライン処理は、図１４（ａ）～図１４（ｄ）に示すパイプライン処理と比較して、命令フェッチステージがＩＦ０およびＩＦ１の２段となっている点、およびデータメモリアクセスステージがＭ０およびＭ１の２段となっている点が異なる。このデータメモリアクセスステージが２段となっているのは、高速データメモリ１０２に

アクセスする場合であり、低電力データメモリ１０４にアクセスする場合にはＭ０～Ｍ２の３段となる。

【００９７】

図１４（ｃ）および図１４（ｇ）に示すように、ＬＤ／ＳＴ命令の実行において、ステージＭ０～Ｍ２またはＭ０～Ｍ１の複数のパイプラインステージでデータメモリにアクセスする場合、データバスにおけるデータ転送の競合が発生するため、たとえばロード命令の実行直後にストア命令の実行ができないという問題点が従来あった。しかし、本実施の形態におけるコア１００は、ロード命令実行時にデータ入力バス１０８を介してデータを転送し、ストア命令実行時にデータ出力バス１０７を介してデータを転送するので、ロード命令の実行直後にストア命令を実行する場合であってもパイプラインが乱れることはない。また、それぞれのデータバスはデータの転送方向が一定であるので、回路が簡単になるというメリットがある。

【００９８】

上述したように、コア１００が高速メモリ１０１または１０２にアクセスする場合と、低電力メモリ１０３または１０４にアクセスする場合とで、パイプライン処理の段数を可変にしている。低電力命令メモリ１０３からフェッチした命令を実行する場合のパイプライン処理におけるタイミングについては、図８を用いて説明した。

【００９９】

図１５は、高速命令メモリ１０１からフェッチした命令を実行する場合のパイプライン処理を説明するためのタイミングチャートである。命令フェッチステージＩＦ０において、最初の１／２クロックで命令フェッチのためのアドレスバス転送と、プリチャージが同時に行なわれる。このプリチャージは、アドレスの値にかかわらず高速命令メモリ１０１にアクセスする際に必ず行なわれるため、低電力命令メモリ１０３にアクセスする場合と比較して、１クロックサイクルだけ早くアクセスが終了する。

【０１００】

そして、次の１／２クロックと命令フェッチステージＩＦ１の最初の１／２ク

ロックとで命令のフェッチが行なわれる。なお、命令フェッチステージＩＦ１において、次の命令をフェッチするために次の命令のアドレスバス転送が行なわれ、パイプライン的に処理が行なわれる。

【０１０１】

命令デコードステージＤにおいて、フェッチされた命令がデコードされる。フェッチされた命令がロード命令である場合、リードレジスタステージＲにおいてレジスタからデータアドレスが読出される。次に、データメモリアクセスステージＭ０において、データアドレスバス転送と同時に高速データメモリ１０２に対するプリチャージが行なわれる。このプリチャージは、アドレスの値にかかわらず高速データメモリ１０２にアクセスする際に必ず行なわれるため、低電力データメモリ１０４にアクセスする場合と比較して、１クロックサイクルだけ早くアクセスが終了する。

【０１０２】

ステージＭ０の次の１／２クロックおよびＭ１において、データの読出しおよびデータバス転送が行なわれる。そして、ライトバックステージＷにおいて、読出したデータがレジスタに書込まれる。

【０１０３】

図１６は、ロード／ストア命令、データ転送命令および比較命令の一覧を示す図である。図１７は、算術演算命令、論理演算命令、シフト演算命令およびビット演算命令の一覧を示す図である。図１８は、分岐命令、ＯＳ（Operating System）関連命令、ＤＳＰ（Digital Signal Processor）関連命令、リピート命令およびデバッガ支援命令の一覧を示す図である。図１６～図１８において、大文字で各サブ命令のニーモニックが記載され、その横にサブ命令のオペレーションの内容が記載されている。各サブ命令のオペレーションの内容は、図１６～図１８に詳細に記載されているので、詳細な説明は行なわない。

【０１０４】

図１９は、本実施の形態におけるプロセッサ１０のメモリマップの一例を示す図である。図１９に示すように、各メモリはアドレス値によって区別されており、コア１００はアドレス値によってメモリアクセス開始からメモリアクセス終了

までのサイクル数、すなわちパイプラインの段数を決定する。なお、図２に示すように、高速命令メモリ１０１および低電力命令メモリ１０３からの命令のフェッチと、高速データメモリ１０２および低電力データメモリ１０４に対するアクセスとが異なるバスで行なわれるため、バスの競合が発生することはない。

【０１０５】

デジタル信号処理においては、ＦＩＲ（Finite Impulse Response）フィルタ等の処理を行なう際にループが多用される。本実施の形態のコア１００においては、ループ処理をハードウェアで行ない、ループのための分岐を実質的にゼロクロックサイクルで処理するためのリピート命令（ＲＥＰＥＡＴ０，ＲＥＰＥＡＴ１）をサポートしている。以下、このリピート命令について詳細に説明する。

【０１０６】

図２０は、ＲＥＰＥＡＴ１のオペレーションとそれを用いたＦＩＲフィルタ処理のプログラムの一例とを示す図である。ＲＥＰＥＡＴ１のオペレーションに示すように、ループの回数ｃｏｕｎｔと、ループの最初の命令から最後の命令までの距離ｐｃａｄｄｒとが指定され、ＲＰＴ１＿Ｃ、ＲＰＴ１＿Ｓ、ＲＰＴ１＿ＥおよびＲＰＴ１＿Ｉ（０：５）の各レジスタに、それぞれループ回数、ループ開始アドレス（ＲＥＰＥＡＴ１命令の直後の命令アドレス）、ループ終了アドレス、およびループの先頭から６個の命令が入力され、ＰＳＷのＲＰ１ビットおよびＦＳ１ビットがセットされる。そして、ループの回数ｃｏｕｎｔだけ、ループの最初の命令から最後の命令までが実行される。

【０１０７】

図２０に示すプログラム例においては、ループの回数ｃｏｕｎｔに“２０”が設定され、ループの最初の命令から最後の命令までの距離ｐｃａｄｄｒに“４８”が設定される。そして、ＳＴＡＲＴからＥＮＤまでの処理、すなわちＬＤ２Ｗ命令とＭＡＣ０命令を並列に実行するＶＬＩＷ命令６個からなる処理が２０回繰返される。

【０１０８】

図２１は、図２０に示すプログラムを実行したときのパイプライン処理を説明するための図である。高速命令メモリ１０１からＬＤ２Ｗ命令およびＭＡＣ０命

令がフェッチされ、メモリ演算部１３０および整数演算部１４０が、ＬＤ２Ｗ命令およびＭＡＣ０命令を並列にパイプライン処理し、１クロックサイクルに１回ずつ積和演算結果がアキュムレータＡ０に格納されるところを示している。

【０１０９】

ＲＥＰＥＡＴ１命令実行直後の１回目のループにおいてはＦＳ１ビットがセットされ、ＳＴＡＲＴからＥＮＤまでのＶＬＩＷ命令が実行されるとともに、その６個のＶＬＩＷ命令が６個のレジスタＲＰＴ１＿Ｉ（０：５）に書込まれる。１回目のループの最後の分岐においてＦＳ１ビットがリセットされ、２回目のループ以降最後のループまで、レジスタＲＰＴ１＿Ｉ（０：５）に格納されたＶＬＩＷ命令がフェッチされて高速命令メモリ１０１へのアクセスは行なわれない。ループの実行回数に応じて、レジスタＲＰＴ１＿Ｃの値が１ずつデクリメントされ、ＲＰＴ１＿Ｃの値が“０”になるとループが終了してＲＰ１ビットがリセットされる。

【０１１０】

また、ループ実行中にプロセッサ１０が割込み要求を受付けると、このプログラムの実行が中断され、現タスクＡが新タスクＢにスイッチするが、タスクＢを実行する前にＯＳがＢＰＳＷに退避されたタスクＡ実行中のＰＳＷ値と、ＲＴＰ１＿Ｃ、ＲＴＰ１＿Ｓ、ＲＴＰ１＿ＥおよびＲＴＰ１＿Ｉ（０：５）の各レジスタを含むタスクＡの実行環境をすべてメモリに退避し、タスクＡの処理に戻るときにＯＳがタスクＡの実行環境を復帰させる。そのため、ＲＥＰＥＡＴ１命令によって起動されたループ処理がタスクスイッチの切替えによって壊れることはない。

【０１１１】

図２２は、ＲＥＰＥＡＴ０のオペレーションとそれを用いたＦＩＲフィルタ処理のプログラムの一例とを示す図である。ＲＥＰＥＡＴ０のオペレーションに示すように、ループの回数ｃｏｕｎｔと、ループの最初の命令から最後の命令までの距離ｐｃａｄｄｒとが指定され、ＲＰＴ０＿Ｃ、ＲＰＴ０＿Ｓ、ＲＰＴ０＿ＥおよびＲＰＴ０＿Ｉの各レジスタに、それぞれループ回数、ループ開始アドレス（ＲＥＰＥＡＴ０命令の直後の命令アドレス）、ループ終了アドレス、およびル

ープの先頭命令が入力され、ＰＳＷのＲＰ０ビットおよびＦＳ０ビットがセットされる。そして、ループの回数ｃｏｕｎｔだけ、ループの最初の命令から最後の命令までが実行される。

【０１１２】

図２２に示すプログラム例においては、ループの回数ｃｏｕｎｔに“１０”が設定され、ループの最初の命令から最後の命令までの距離ｐｃａｄｄｒに“６４”が設定される。そして、ＳＴＡＲＴからＥＮＤまでの処理、すなわちＬＤ２Ｗ命令とＭＡＣ０命令とを並列に実行するＶＬＩＷ命令８個からなる処理が１０回繰返される。

【０１１３】

図２３は、図２２に示すプログラムを実行したときのパイプライン処理を説明するための図である。低電力命令メモリ１０３からＬＤ２Ｗ命令およびＭＡＣ０命令がフェッチされ、メモリ演算部１３０および整数演算部１４０が、ＬＤ２Ｗ命令およびＭＡＣ０命令を並列にパイプライン処理し、１クロックサイクルに１回ずつ積和演算結果がアキュムレータＡ０に格納されるところを示している。

【０１１４】

ＲＥＰＥＡＴ０命令実行直後の１回目のループにおいてはＦＳ０ビットがセットされ、ＳＴＡＲＴからＥＮＤまでのＶＬＩＷ命令が実行されるとともに、最初のＶＬＩＷ命令（ラベルＳＴＡＲＴが付されたＶＬＩＷ命令）がレジスタＲＰＴ０＿Ｉに書込まれる。１回目のループの最後の分岐においてＦＳ０ビットがリセットされ、２回目のループ以降最後のループまで、レジスタＲＰＴ０＿Ｉに格納された最初のＶＬＩＷ命令および低電力命令メモリ１０３に格納された最初のＶＬＩＷ命令以外の命令がフェッチされて実行される。ループの実行回数に応じて、レジスタＲＰＴ０＿Ｃの値が１ずつデクリメントされ、ＲＰＴ０＿Ｃの値が“０”になるとループが終了してＲＰ０ビットがリセットされる。

【０１１５】

図２２に示す８個のＶＬＩＷ命令が低電力命令メモリ１０３のメモリバンク４０～４７に保持されており、１回目のループにおいてはメモリバンク４０から順番にＶＬＩＷ命令がアクセスされて、１クロックサイクル毎にＶＬＩＷ命令がフ

ェッチされる。図２２に示すプログラム例の場合には、ＶＬＩＷ命令が８個であるので連続して同じメモリバンクにアクセスすることがなく、スループットが１となる。しかし、ループ内のＶＬＩＷ命令の数が（バンク数×ｎ＋１）の場合には、ループ内の最後の命令と最初の命令とが同じメモリバンクに存在することになり、１ウェイト挿入されてスループットが２となる。

【０１１６】

一方、本実施の形態におけるプロセッサ１０においては、ループの最初の命令がレジスタＲＰＴ０＿Ｉに保持されているため、連続して同じメモリバンクにアクセスされることがなくなり、図２３に示すようにパイプラインに乱れが発生することはない。

【０１１７】

図２４は、図２２に示すプログラムのループ部分が実行されるときのデータの流れを示す図である。メモリ演算部１３０および整数演算部１４０は、ＬＤ２Ｗ命令およびＭＡＣ０命令を並列に実行する。メモリ演算部１３０は、低電力データメモリ１０４の係数領域３０２（アドレスH’ 8000 8100～H’ 8000 8128）からレジスタＲ１０～Ｒ１７へ係数を２個ずつロードするＬＤ２Ｗ命令と、低電力データメモリ１０４の変数領域３０１（アドレスH’ 8000 0100～H’ 8000 0128）からレジスタＲ２０～Ｒ２７へ変数を２個ずつロードするＬＤ２Ｗ命令とが交互に実行される。

【０１１８】

また、メモリ演算部１３０は、レジスタＲ３０またはＲ３１に保持しているポインタを命令実行毎に８ずつポストインクリメントし、連続する係数および変数を次々にレジスタファイル１２０へロードする。メモリ演算部１３０は、奇数番目のＬＤ２Ｗ命令を実行することによって、メモリバンク６０～６３に保持される係数データをレジスタＲ１０～Ｒ１７にロードする。また、メモリ演算部１３０は、偶数番目のＬＤ２Ｗ命令を実行することによって、メモリバンク６４～６７に保持される変数データをレジスタＲ２０～Ｒ２７にロードする。

【０１１９】

図１１に示すように、低電力データメモリ１０４のメモリバンク６０～６７の

ＣＳ信号を生成する際に、Ａ１６、Ａ２７およびＡ２８をデコードしている。したがって、低電力データメモリ１０４のメモリバンク６０～６３が変数領域３０１となり、メモリバンク６４～６７が係数領域３０２となるため、図２２に示すＬＤ２Ｗ命令を連続して実行した場合であっても、同じメモリバンクにアクセスされることがなくパイプラインに乱れが発生することはない。

【０１２０】

また、たとえ連続して係数をアクセスする場合や、連続して変数をアクセスする場合であっても、Ａ２７およびＡ２８をデコードしてメモリバンクのＣＳ信号を生成しているため、連続する係数や変数が同じメモリバンクに存在することは起こり得ない。したがって、ポストインクリメントアドレッシングモードを使用する限り、同じバンクへ連続してアクセスされることはない。

【０１２１】

本実施の形態においては、低電力命令メモリ１０３の同じメモリバンクに連続アクセスが行なわれないプログラム例として、ＦＩＲフィルタ処理について説明した。しかし、命令のフェッチにおいて分岐が発生しない限り連続するアドレスにアクセスが行なわれる。したがって、下位アドレスをデコードして２つ以上のメモリバンクに分割すれば、分岐以外の命令実行において同じメモリバンクに連続してアクセスが発生することはない。

【０１２２】

また、デジタル信号処理において、ＦＩＲフィルタ処理以外でも連続したアドレス領域に係数や変数を格納して順次アクセスする場合、下位アドレスをデコードして低電力データメモリ１０４を２つ以上のメモリバンクに分割しておけば、同じメモリバンクに連続してアクセスが発生することはない。

【０１２３】

以上説明したように、本実施の形態におけるデータ処理装置によれば、メモリ演算部１３０が低電力命令メモリ１０３のメモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させるので、選択したメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、パイプラインステージＩＦ０～ＩＦ２が並

列に行なわれるため、低電力命令メモリ１０３のスループットを向上させることが可能となった。

【０１２４】

また、バンク選択回路４８は、下位アドレスＡ２７およびＡ２８をデコードしてメモリバンク４０～４７のＣＳ信号を生成するので、メモリ演算部１３０が連続したアドレス領域から命令をフェッチする場合でも異なるメモリバンクにアクセスが行なわれるため、パイプラインに乱れが発生することを防止できる。また、高速命令メモリ１０１から命令をフェッチする場合には、バンクセレクトを行なわず、プリチャージとアドレス転送とを並列して行なうようにしたので、高速に命令をフェッチすることが可能となった。

【０１２５】

メモリ演算部１３０が低電力データメモリ１０４のメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させるので、選択したメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、パイプラインステージＭ０～Ｍ２が並列に行なわれるため、低電力データメモリ１０４のスループットを向上させることが可能となった。

【０１２６】

また、バンク選択回路６８は、下位アドレスＡ２７およびＡ２８をデコードしてメモリバンク６０～６７のＣＳ信号を生成するので、メモリ演算部１３０が連続したアドレス領域にデータのアクセスを行なう場合でも異なるメモリバンクにアクセスが行なわれるため、パイプラインに乱れが発生することを防止できる。また、高速データメモリ１０２にデータのアクセスを行なう場合には、バンクセレクトを行なわず、アドレスの転送とプリチャージとを並列して行なうようにしたので、高速にデータのアクセスを行なうことが可能となった。

【０１２７】

また、バンク選択回路６８は、上位アドレスＡ１６をデコードしてメモリバンク６０～６７のＣＳ信号を生成するので、係数と変数とを別々の領域に格納することによって、係数と変数とを交互に読み出す場合でも同じメモリバンクに連続

してアクセスが発生することがなくなり、パイプラインに乱れが発生することを防止できる。

【０１２８】

メモリ演算部１３０は、リピート命令を実行する場合に、レジスタＲＰＴ０＿ＩまたはＲＰＴ１＿Ｉ（ｎ）にリピート命令の直後の命令を保持するようにしたので、ループの最後の命令からループの最初の命令に分岐する場合であっても、同じメモリバンクに連続してアクセスすることがなくなり、処理性能の低下を防止することが可能となった。

【０１２９】

また、ループ実行中にタスクスイッチが切替えられる場合であっても、ＯＳがＢＰＳＷに退避されたＰＳＷ値と、ＲＴＰ１＿Ｃ、ＲＴＰ１＿Ｓ、ＲＴＰ１＿ＥおよびＲＴＰ１＿Ｉ（０：５）の各レジスタを含むタスクの実行環境をすべてメモリに退避するようにしたので、ＲＥＰＥＡＴ命令によって起動されたループ処理がタスクスイッチの切替えによって壊れるのを防止することが可能となった。

【０１３０】

また、最初のループにおいてＦＳ１ビットがセットされ、２回目以降のループにおいてＦＳ１ビットがリセットされるので、リピート命令の実行状況を容易に把握することが可能となった。

【０１３１】

今回開示された実施の形態は、すべての点で例示であって制限的なものではないと考えられるべきである。本発明の範囲は上記した説明ではなくて特許請求の範囲によって示され、特許請求の範囲と均等の意味および範囲内でのすべての変更が含まれることが意図される。

【０１３２】

【発明の効果】

請求項１に記載のデータ処理装置によれば、メモリ演算部が命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとを発生させるので、選択した命令メモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、命令メモリバ

ンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとが並列に行なわれるため、命令メモリのスループットを向上させることが可能となった。

【０１３３】

請求項２に記載のデータ処理装置によれば、同じ命令メモリバンクに連続してアクセスが行なわれなくなり、パイプラインの乱れを防止することが可能となった。

【０１３４】

請求項３に記載のデータ処理装置によれば、メモリ演算部が高速命令メモリから命令をフェッチする場合に、高速命令メモリに対して少ないパイプラインステージ数でアクセスを行なうので、高速命令メモリのレイテンシを向上させることが可能となった。

【０１３５】

請求項４に記載のデータ処理装置によれば、メモリ演算部がデータメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとを発生させるので、選択したデータメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとが並列に行なわれるため、データメモリのスループットを向上させることが可能となった。

【０１３６】

請求項５に記載のデータ処理装置によれば、変数データと係数データとを交互に読出す場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１３７】

請求項６に記載のデータ処理装置によれば、ポストインクリメントを用いてデータメモリにアクセスする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１３８】

請求項７に記載のデータ処理装置によれば、メモリ演算部が高速データメモリにアクセスする場合に、高速データメモリに対して少ないパイプラインステージ数でアクセスを行なうので、高速データメモリのレイテンシを向上させることが可能となった。

【０１３９】

請求項８に記載のデータ処理装置によれば、命令メモリからの命令のフェッチと、データメモリへのアクセスとのバスの競合が発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１４０】

請求項９に記載のデータ処理装置によれば、データの読出しと、データの書込みとのバスの競合が発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１４１】

請求項１０に記載のデータ処理装置によれば、メモリ演算部が専用レジスタに保持された命令をフェッチしながらリピート命令を実行するので、複数のメモリバンクに分割された命令メモリから命令をフェッチする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、処理性能の向上を図ることが可能となった。

【０１４２】

請求項１１に記載のデータ処理装置によれば、プロセッサ状態ワードのフラグにより、リピート命令の実行状況を制御してマルチタスクにも対応することが可能となった。

【０１４３】

請求項１２に記載のデータ処理装置によれば、メモリ演算部が複数の専用レジスタに保持された複数の命令をフェッチしながらリピート命令を実行するので、複数のメモリバンクに分割された命令メモリから命令をフェッチする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、処理性能の向上を図ることが可能となった。

【０１４４】

請求項１３に記載のデータ処理装置によれば、プロセッサ状態ワードのフラグにより、リピート命令の実行状況を制御してマルチタスクにも対応することが可能となった。

【０１４５】

請求項１４に記載のデータ処理装置によれば、選択した命令メモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、命令メモリバンクの選択に対応したパイプラインステージと、命令の読出しに対応したパイプラインステージとが並列に行なわれるため、命令メモリのスループットを向上させることが可能となった。

【０１４６】

請求項１５に記載のデータ処理装置によれば、同じ命令メモリバンクに連続してアクセスが行なわれなくなり、パイプラインの乱れを防止することが可能となった。

【０１４７】

請求項１６に記載のデータ処理装置によれば、選択したデータメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となった。また、データメモリバンクの選択に対応したパイプラインステージと、データのアクセスに対応したパイプラインステージとが並列に行なわれるため、データメモリのスループットを向上させることが可能となった。

【０１４８】

請求項１７に記載のデータ処理装置によれば、変数データと係数データとを交互に読出す場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１４９】

請求項１８に記載のデータ処理装置によれば、ポストインクリメントを用いてデータメモリにアクセスする場合であっても、同じメモリバンクに連続したアクセスが発生しなくなり、パイプラインの乱れを防止することが可能となった。

【０１５０】

請求項１９に記載のデータ処理装置によれば、元のタスクに復帰する際に、容

易にタスクスイッチを切替えることが可能となった。

【図面の簡単な説明】

【図１】　本発明の実施の形態におけるプロセッサ１０を用いたデータ処理装置の概略構成を示すブロック図である。

【図２】　本発明の実施の形態におけるプロセッサ１０の概略構成を示すブロック図である。

【図３】　コア１００が有するレジスタを説明するための図である。

【図４】　ＰＳＷの詳細を説明するための図である。

【図５】　低電力命令メモリ１０３の概略構成を示すブロック図である。

【図６】　バンク選択回路４８の詳細を説明するためのブロック図である。

【図７】　メモリバンクのＣＳ信号の生成を説明するための図である。

【図８】　メモリバンク４０～４７にアクセスするときのパイプライン処理を説明するための図である。

【図９】　低電力データメモリ１０４の概略構成を示すブロック図である。

【図１０】　バンク選択回路６８の詳細を説明するためのブロック図である。

【図１１】　メモリバンクのＣＳ信号の生成を説明するための図である。

【図１２】　コア１００が実行する命令のフォーマットを説明するための図である。

【図１３】　Ｌコンテナ２０５およびＲコンテナ２０６に保持されるサブ命令のフォーマットを示す図である。

【図１４】　本発明の実施の形態におけるコア１００のパイプライン処理を説明するための図である。

【図１５】　高速命令メモリ１０１からフェッチした命令を実行する場合のパイプライン処理を説明するための図である。

【図１６】　ロード／ストア命令、データ転送命令および比較命令の一覧を示す図である。

【図１７】　算術演算命令、論理演算命令、シフト演算命令およびビット演算命令の一覧を示す図である。

【図１８】　分岐命令、ＯＳ関連命令、ＤＳＰ関連命令、リピート命令およびデバッガ支援命令の一覧を示す図である。

【図１９】　本発明の実施の形態におけるプロセッサ１０のメモリマップの一例を示す図である。

【図２０】　ＲＥＰＥＡＴ１命令のオペレーションとそれを用いたＦＩＲフィルタ処理のプログラムの一例とを示す図である。

【図２１】　図２０に示すプログラムを実行したときのパイプライン処理を説明するための図である。

【図２２】　ＲＥＰＥＡＴ０命令のオペレーションとそれを用いたＦＩＲフィルタ処理のプログラムの一例とを示す図である。

【図２３】　図２２に示すプログラムを実行したときのパイプライン処理を説明するための図である。

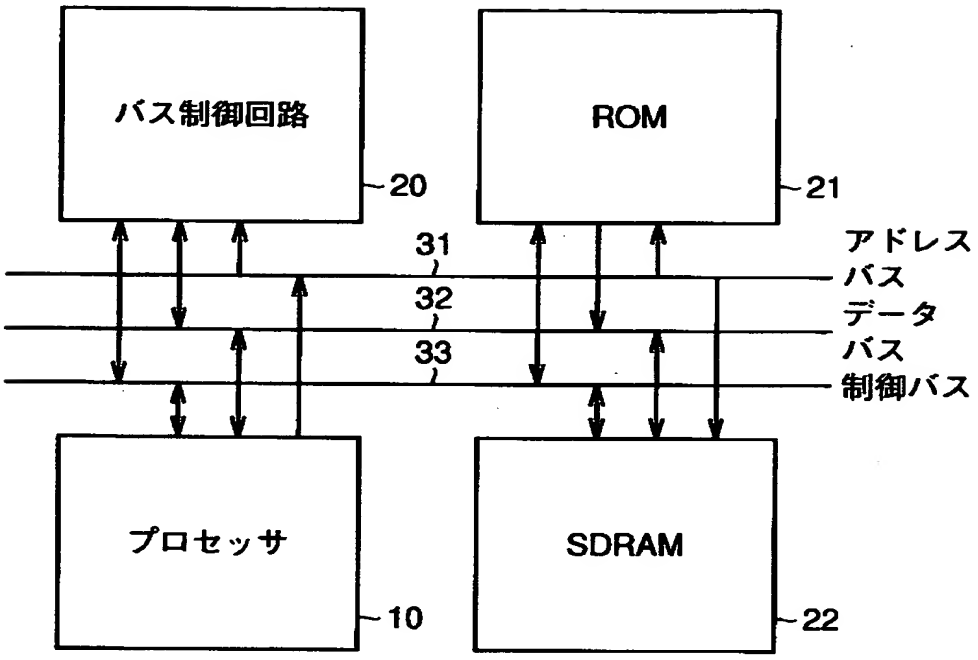
【図２４】　図２２に示すプログラムのループ部分が実行されるときのデータの流れを示す図である。

【符号の説明】

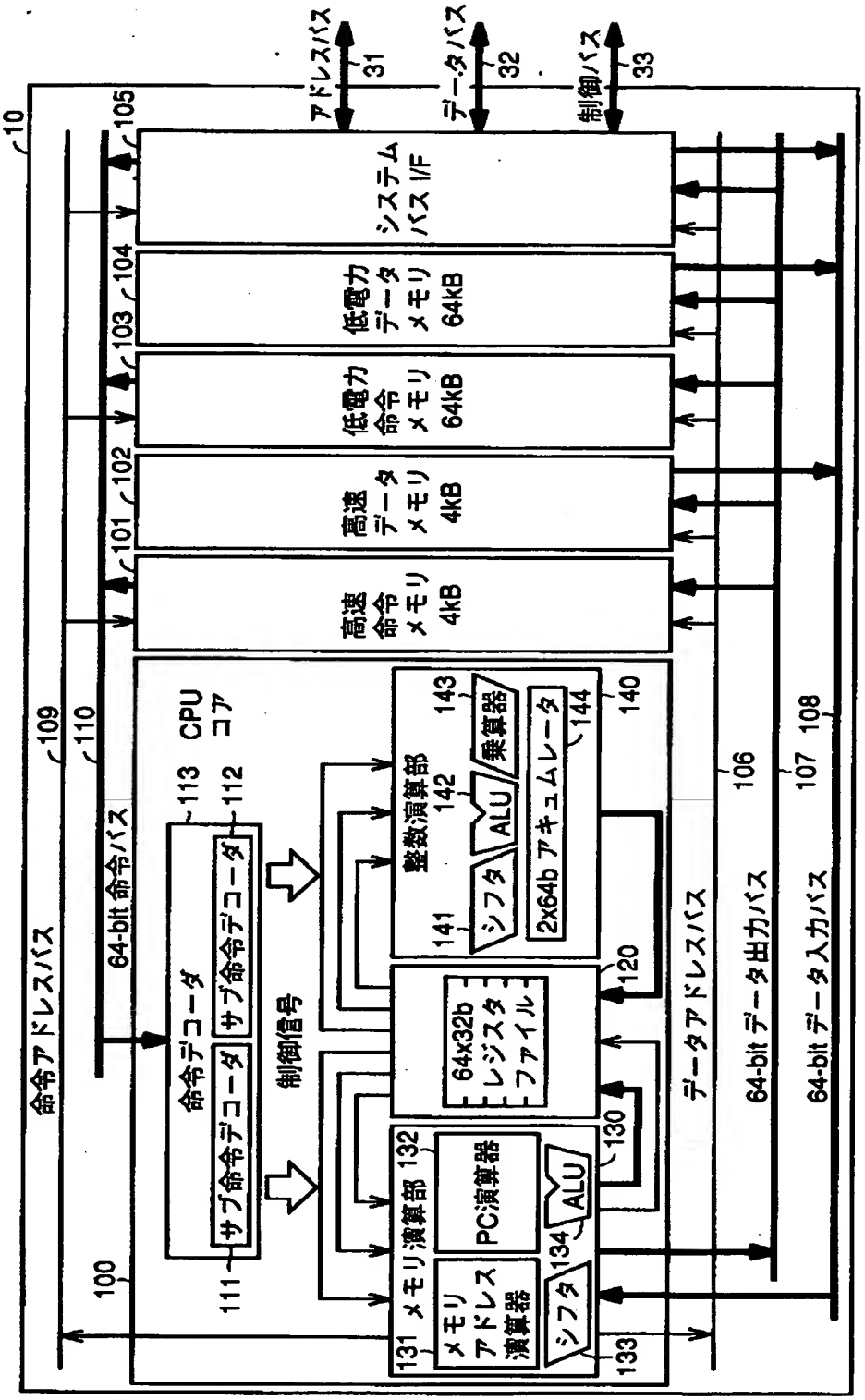
１０　プロセッサ、２０　バス制御回路、２１　ＲＯＭ、２２　ＳＤＲＡＭ、４０～４７，６０～６７　メモリバンク、４８，６８　バンク選択回路、５１，７１　アドレス入力レジスタ、５２，７２　データ入力レジスタ、５３　命令出力レジスタ、５４，７４　ＣＳ信号生成回路、５５　マルチプレクサ、５６，５７　ラッチ、７３　データ出力レジスタ、１００　ＣＰＵコア、１０１　高速命令メモリ、１０２　高速データメモリ、１０３　低電力命令メモリ、１０４　低電力データメモリ、１１１，１１２　サブ命令デコーダ、１１３　命令デコーダ、１２０　レジスタファイル、１３０　メモリ演算部、１３１　メモリアドレス演算器、１３２　ＰＣ演算器、１３３，１４１　シフタ、１３４，１４２　ＡＬＵ、１４０　整数演算部、１４３　乗算器、１４４　アキュムレータ。

【書類名】　図面

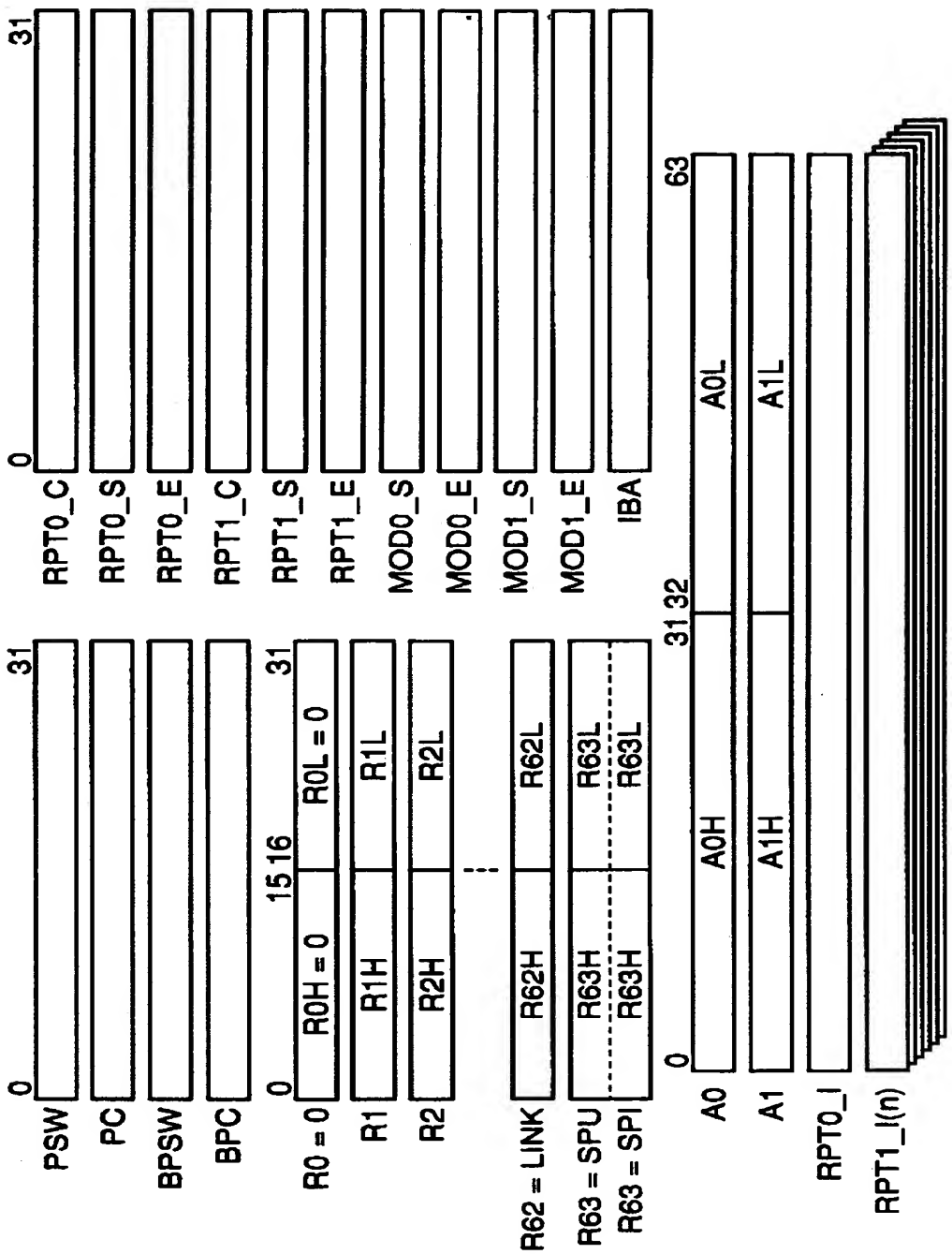
【図１】



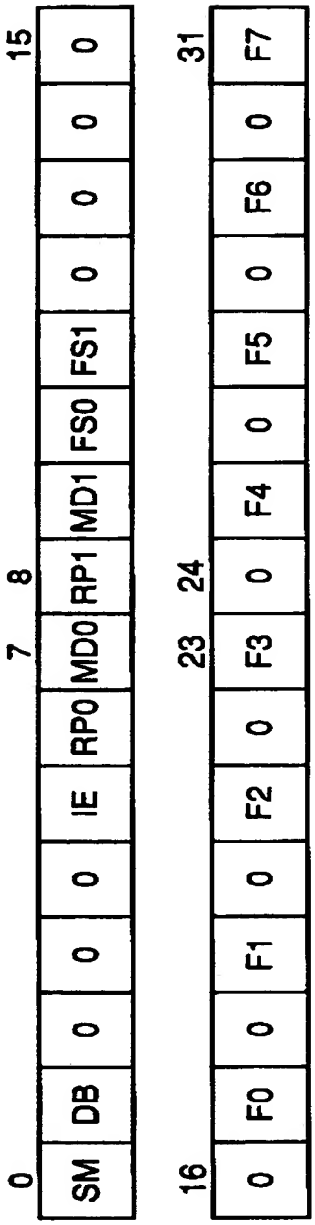
【図２】


アドレスバス
31
データバス
32
制御バス
33
10
105
システム バス I/F
104
低電力 データ メモリ 64kB
103
低電力 命令 メモリ 64kB
102
高速 データ メモリ 4kB
101
高速 命令 メモリ 4kB
109
110
CPU コア
113
112
命令デコーダ
サブ命令デコーダ
サブ命令デコーダ
111
64-bit 命令バス
命令アドレスバス
制御信号
整数演算部
142
143
シフタ
ALU
乗算器
141
2x64b アキュムレータ
144
140
64x32b レジスタ ファイル
120
メモリ演算部 132
PC演算器
ALU
130
メモリ アドレス 演算器
シフタ
134
131
133
100
データアドレスバス
106
64-bit データ出力バス
107
108
64-bit データ入力バス

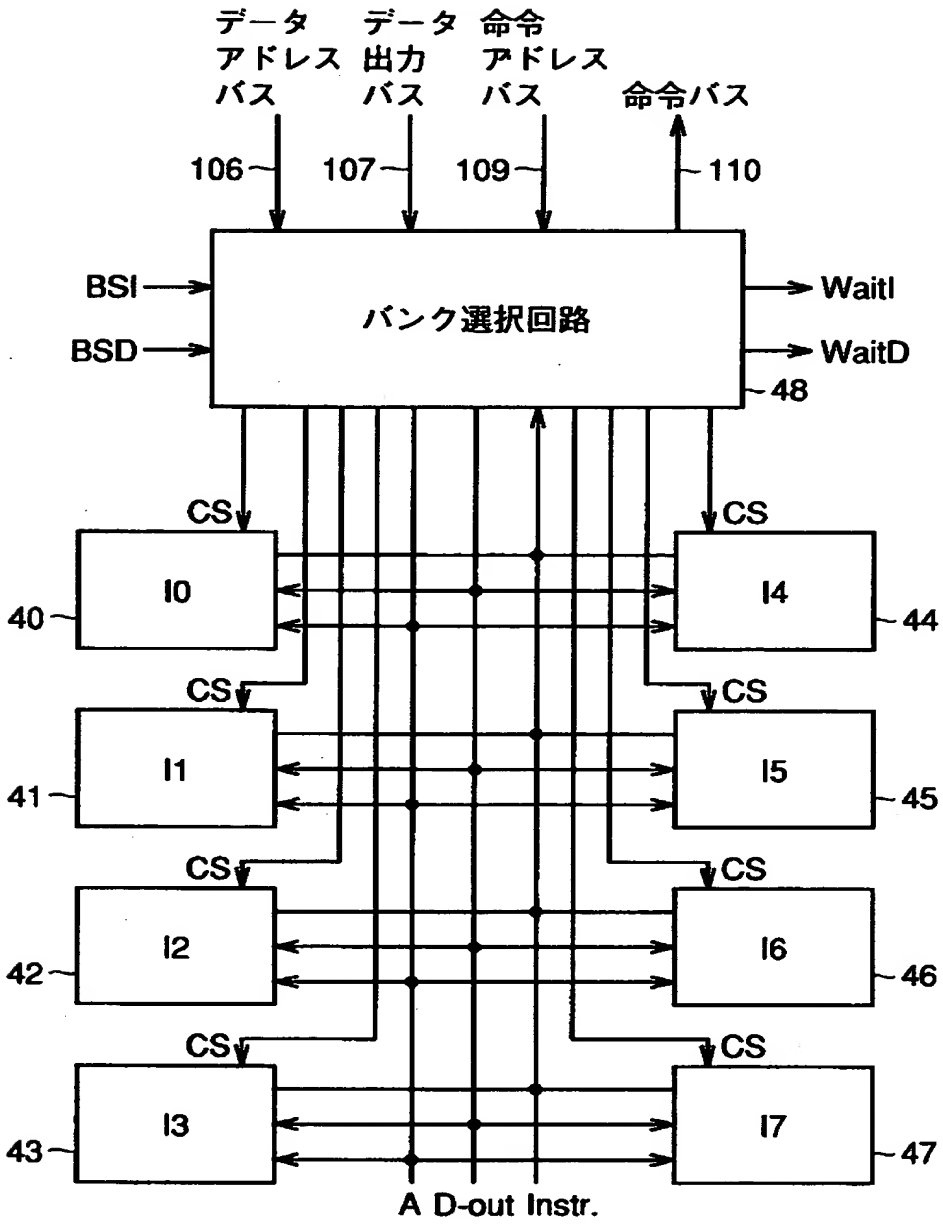
【図３】


31
0
RPT0_C
RPT0_S
RPT0_E
RPT1_C
RPT1_S
RPT1_E
MOD0_S
MOD0_E
MOD1_S
MOD1_E
IBA
31
0
PSW
PC
BPSW
BPC
31
15 16
0
R0H = 0
R0L = 0
R1H
R1L
R2H
R2L
R62H
R62L
R63H
R63L
R63H
R63L
R0 = 0
R1
R2
R62 = LINK
R63 = SPU
R63 = SPI
63
31 32
0
A0H
A0L
A1H
A1L
A0
A1
RPT0_I
RPT1_I(n)

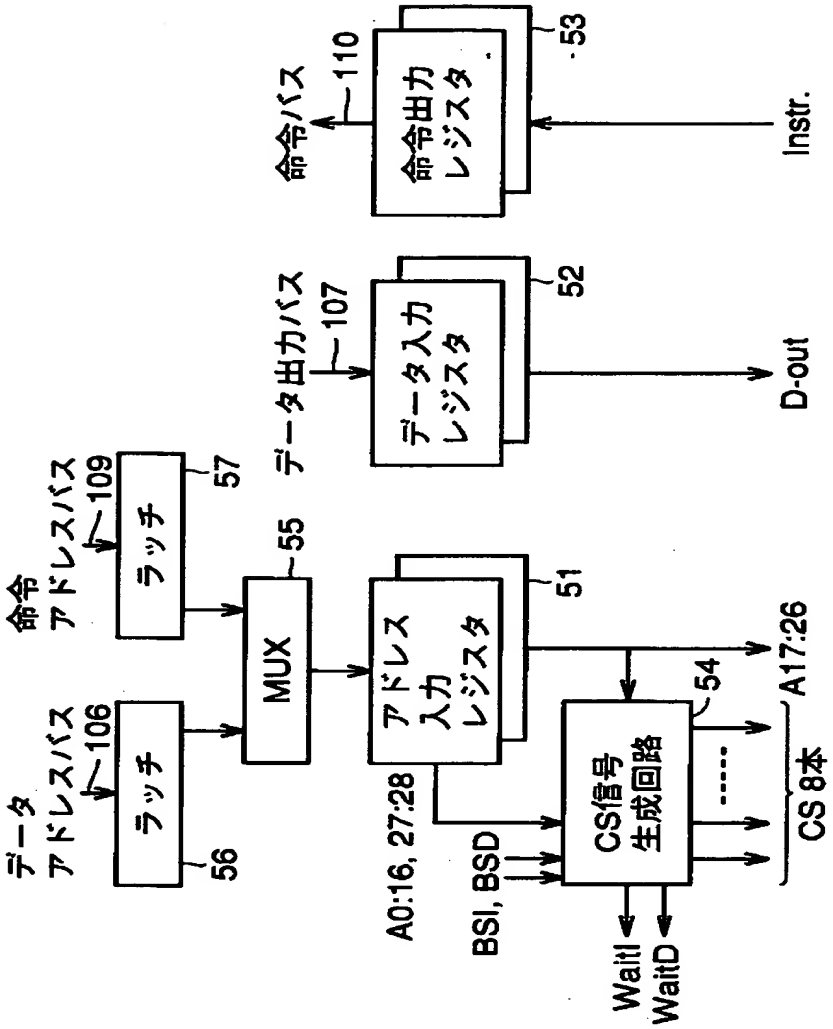
【図４】


15
0
0
0
0
FS1
FS0
MD1
RP1
MD0
RP0
IE
0
0
DB
SM
8
7
0
31
F7
0
F6
0
F5
0
F4
0
F3
0
F2
0
F1
0
F0
0
24
23
16

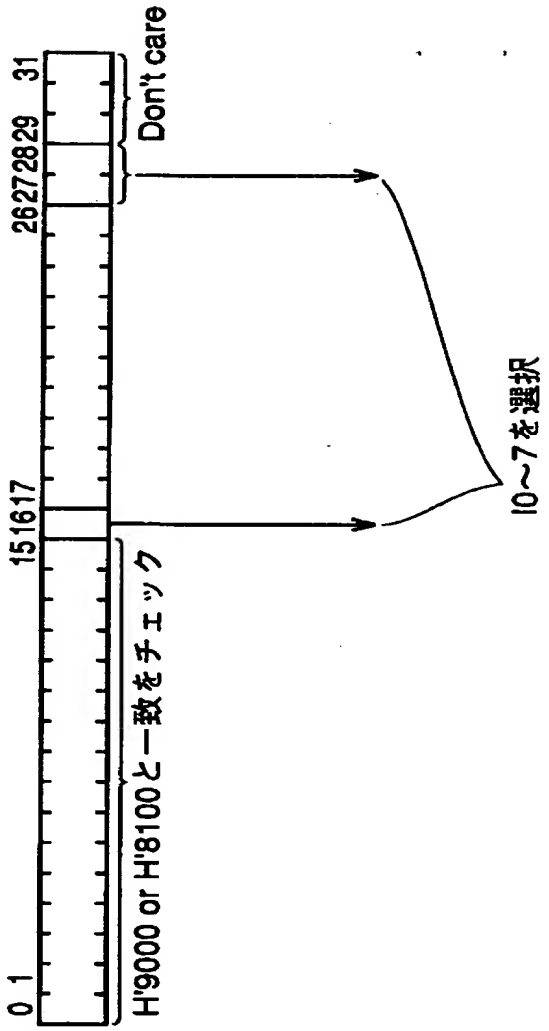
【図５】


データ
アドレス
バス
データ
出力
バス
命令
アドレス
バス
命令バス
106
107
109
110
BSI
BSD
バンク選択回路
WaitI
WaitD
48
CS
I0
40
I1
41
I2
42
I3
43
I4
44
I5
45
I6
46
I7
47
A D-out Instr.

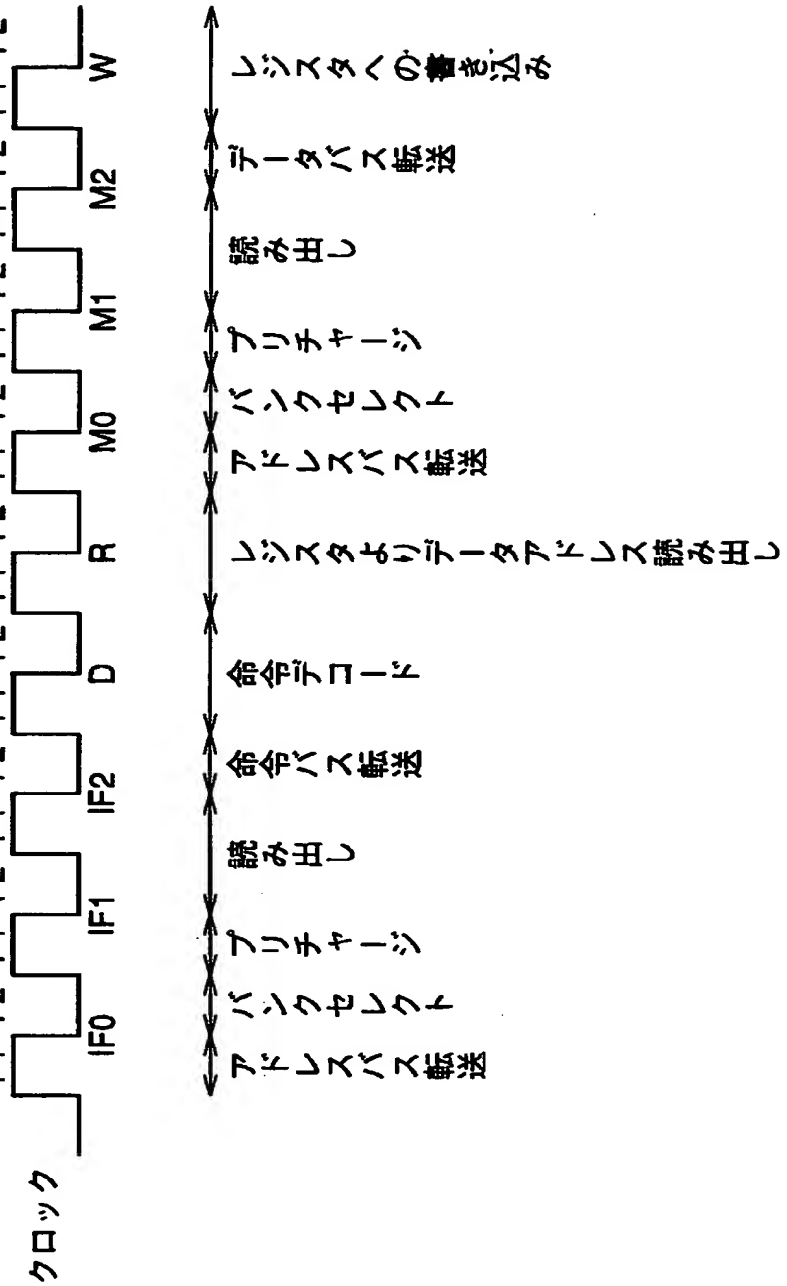
【図６】


命令バス
110
命令出力レジスタ
53
Instr.
データ出力バス
107
データ入力レジスタ
52
D-out
命令アドレスバス
109
ラッチ
57
データアドレスバス
106
ラッチ
56
MUX
55
アドレス入力レジスタ
51
A17:26
54
CS信号生成回路
CS 8本
A0:16, 27:28
BSI, BSD
WaitI
WaitD

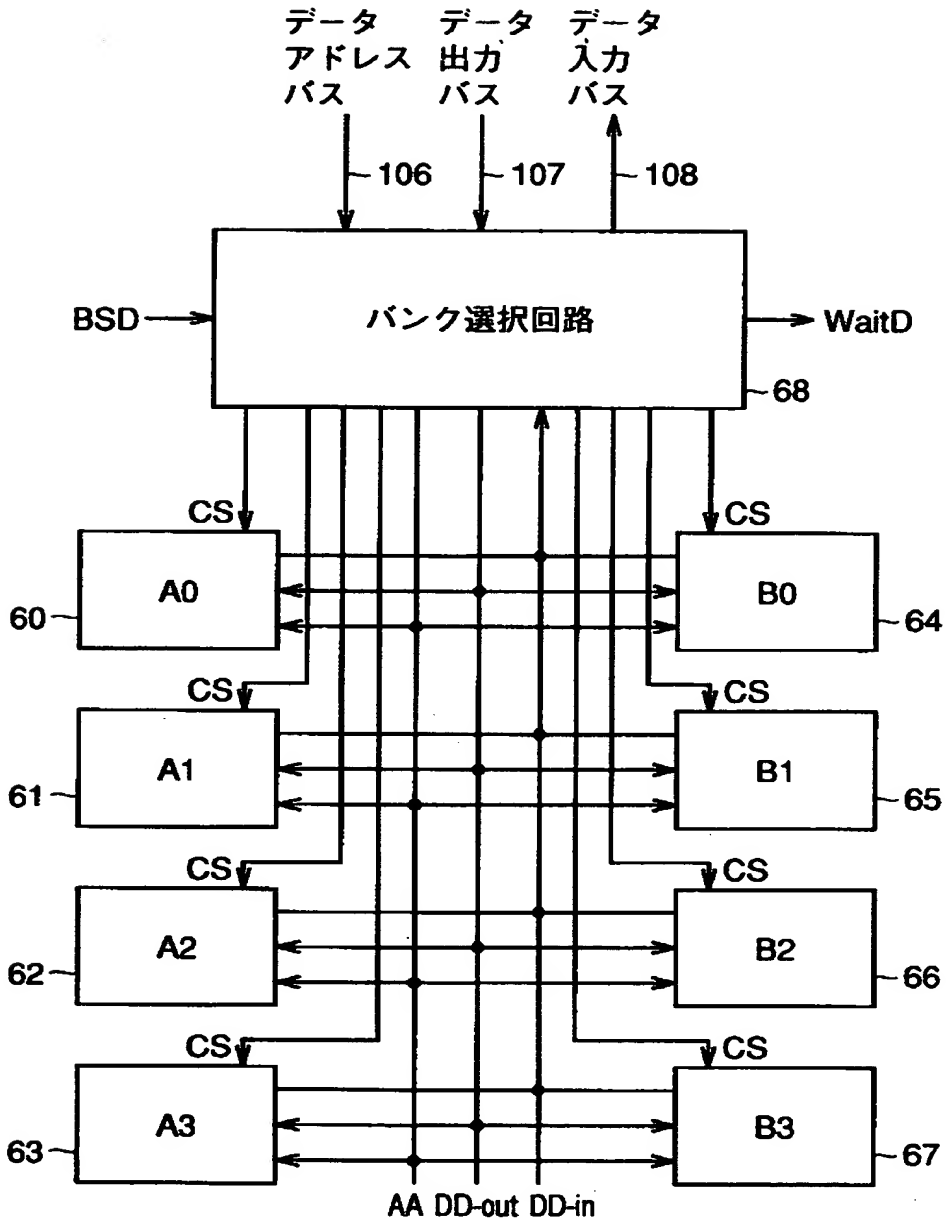
【図７】


31
29
28
27
26
17
16
15
1
0
Don't care
H'9000 or H'8100と一致をチェック
10～7を選択

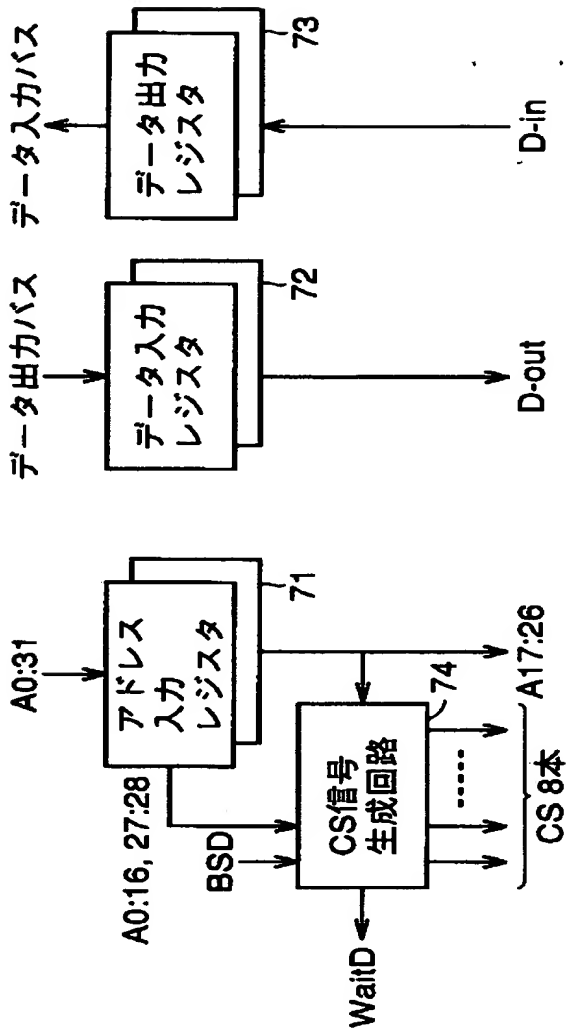
【図８】


クロック
P1
P2
IF0
IF1
IF2
D
R
M0
M1
M2
W
アドレスバス転送
バスクセレクト
プリチャージ
読み出し
命令バス転送
命令デコード
レジスタよりデータアドレス読み出し
アドレスバス転送
バスクセレクト
プリチャージ
読み出し
データバス転送
レジスタへの書き込み

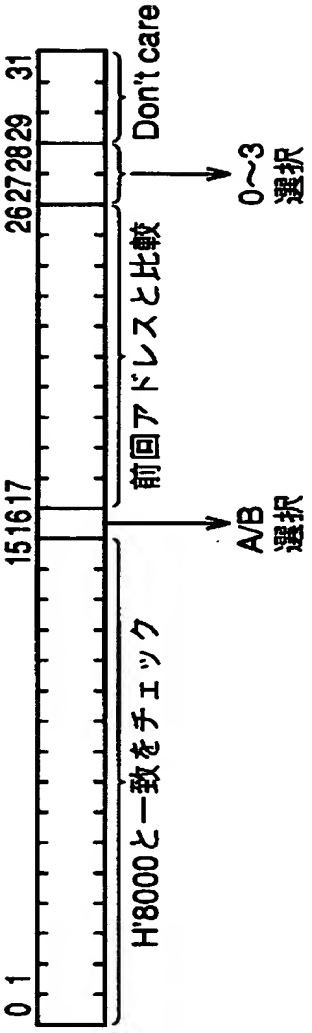
【図９】


データ
アドレス
バス
データ
出力
バス
データ
入力
バス
106
107
108
BSD
バンク選択回路
WaitD
68
CS
A0
60
B0
64
A1
61
B1
65
A2
62
B2
66
A3
63
B3
67
AA DD-out DD-in

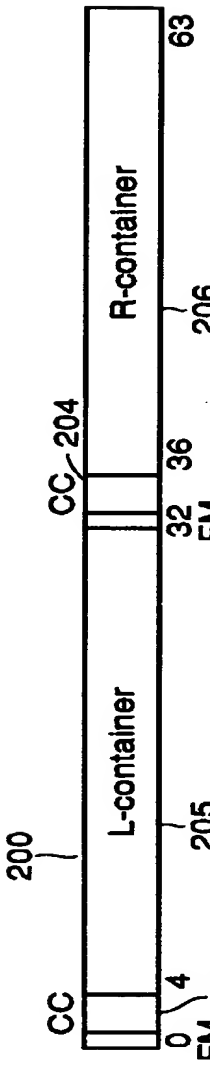
【図１０】


データ入力バス
データ出力レジスタ
73
D-in
データ出力バス
データ入力レジスタ
72
D-out
A0:31
アドレス入力レジスタ
71
A17:26
74
CS信号生成回路
CS 8本
A0:16, 27:28
BSD
WaitD

【図１１】

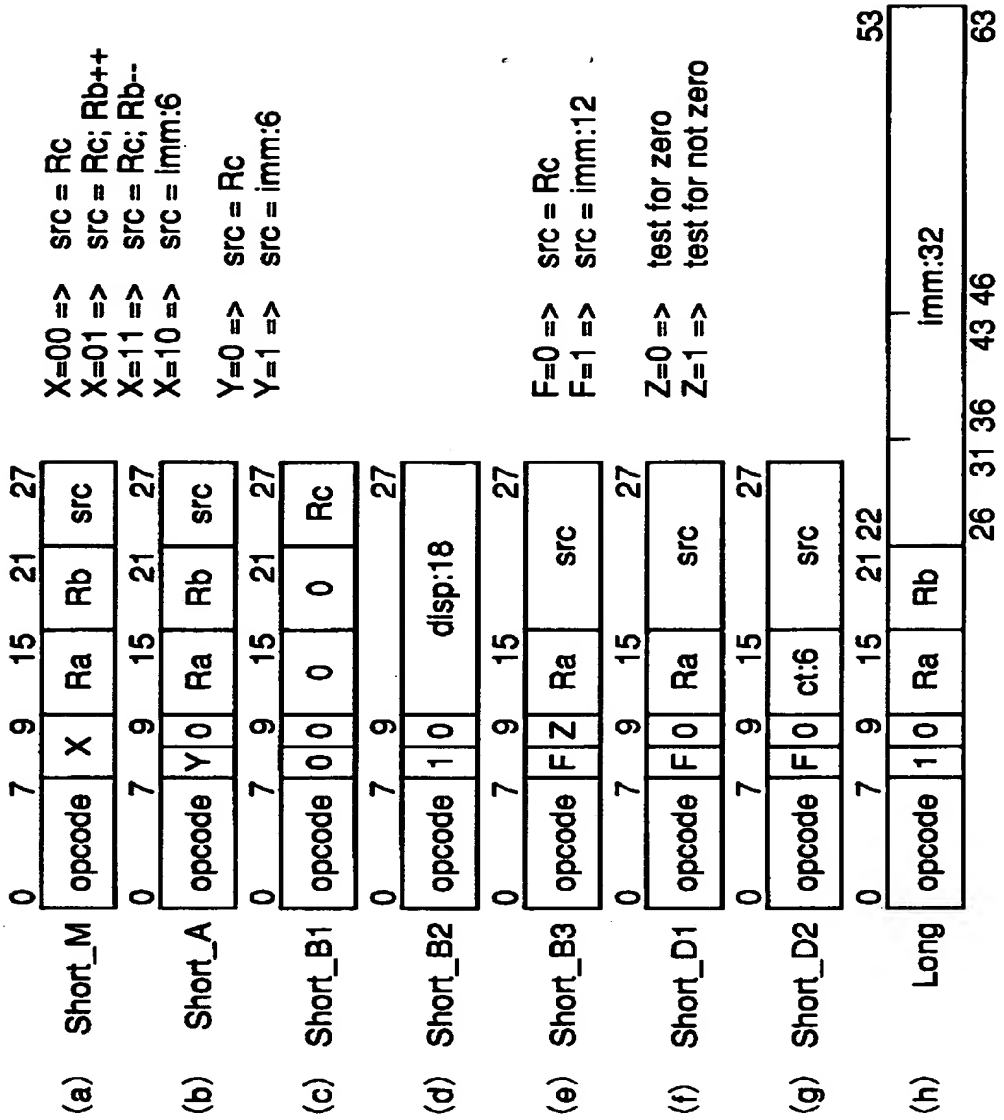

31
2627282 9
Don't care
0~3
選択
前回アドレスと比較
151617
A/B
選択
H'8000と一致をチェック
0 1

【図１２】

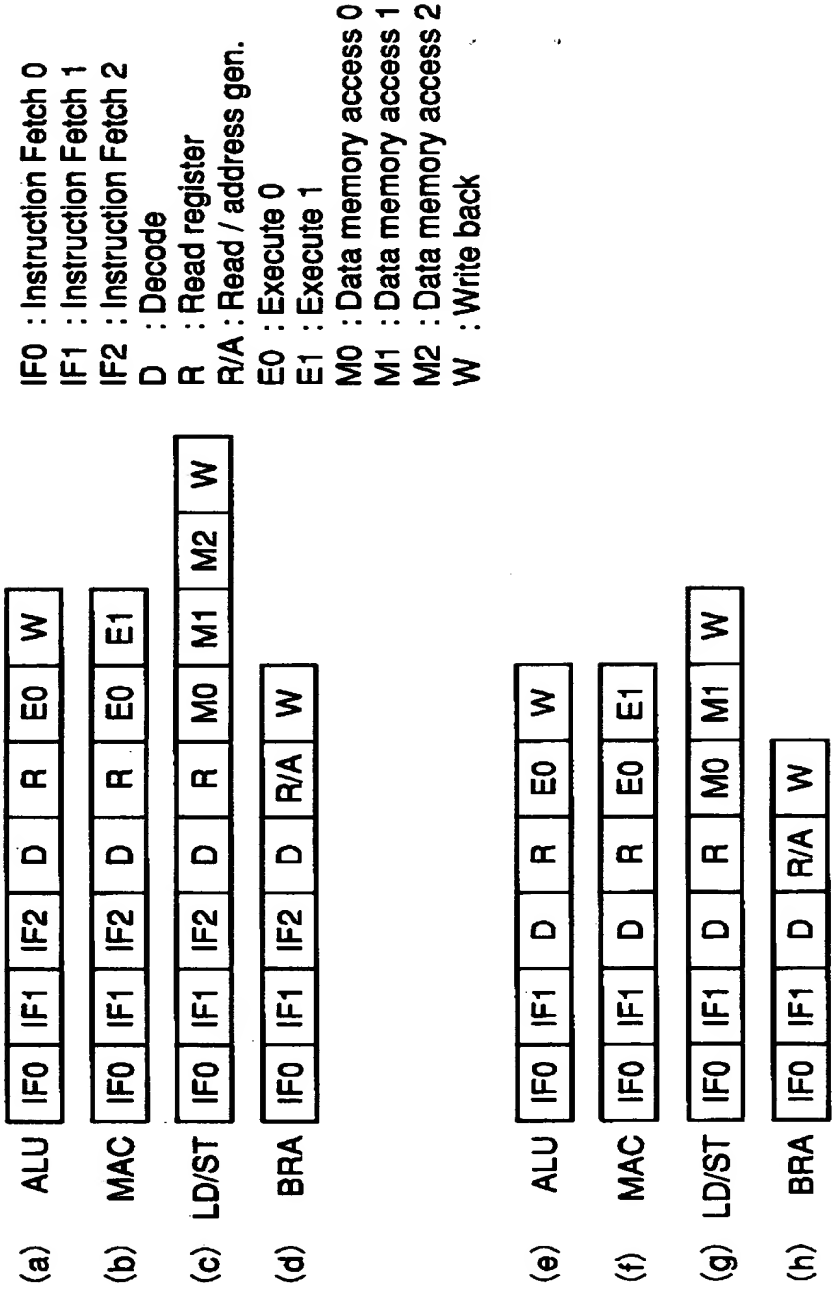


FM	Number of sub-instructions	Issuing Order	
		L-container	R-container
00	two	1st	1st
01	two	1st	2nd
10	two	2nd	1st
11	one	1st	---

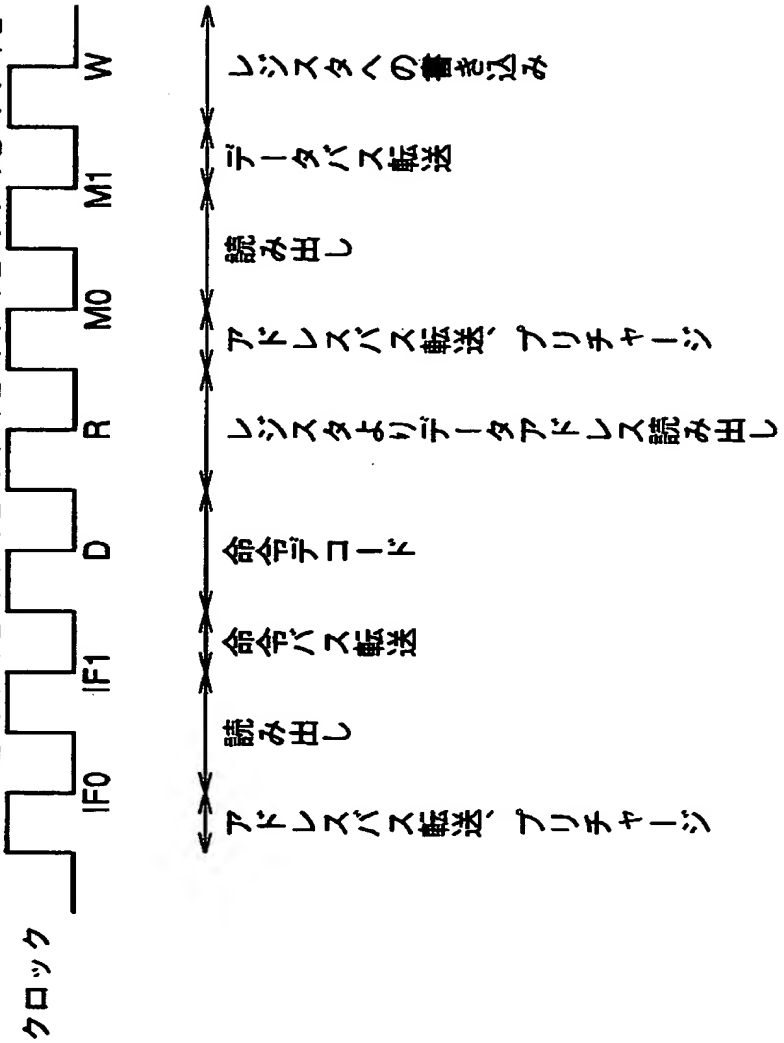
【図１３】


(a) Short_M
(b) Short_A
(c) Short_B1
(d) Short_B2
(e) Short_B3
(f) Short_D1
(g) Short_D2
(h) Long
opcode
X
Y
F
Z
Ra
Rb
src
Rc
disp:18
ct:6
imm:32
0
7
9
15
21
22
26
27
31
36
43
46
53
63
X=00 => src = Rc
X=01 => src = Rc; Rb++
X=11 => src = Rc; Rb--
X=10 => src = imm:6
Y=0 => src = Rc
Y=1 => src = imm:6
F=0 => src = Rc
F=1 => src = imm:12
Z=0 => test for zero
Z=1 => test for not zero

【図１４】


IF0 : Instruction Fetch 0
IF1 : Instruction Fetch 1
IF2 : Instruction Fetch 2
D : Decode
R : Read register
R/A : Read / address gen.
E0 : Execute 0
E1 : Execute 1
M0 : Data memory access 0
M1 : Data memory access 1
M2 : Data memory access 2
W : Write back
(a) ALU IF0 IF1 IF2 D R E0 W
(b) MAC IF0 IF1 IF2 D R E0 E1
(c) LD/ST IF0 IF1 IF2 D R M0 M1 M2 W
(d) BRA IF0 IF1 IF2 D R/A W
(e) ALU IF0 IF1 D R E0 W
(f) MAC IF0 IF1 D R E0 E1
(g) LD/ST IF0 IF1 D R M0 M1 W
(h) BRA IF0 IF1 D R/A W

【図１５】


クロック
P1
P2
IF0
IF1
D
R
M0
M1
W
命令デコード
命令バス番送
読み出し
アドレスバス番送，プリチャージ
データバス番送
レジスタへの書き込み
レジスタよりデータアドレス読み出し

【図１６】

- Load/Store instructions

LDB	Load one byte to a register with sign extension
LDBU	Load one byte to a register with zero extension
LDH	Load one half-word to a register with sign extension
LDHH	Load one half-word to a register high
LDHU	Load one half-word to a register with zero extension
LDW	Load one word to a register
LD2W	Load two words to registers
LD4BH	Load four bytes to four half-word registers with sign extension
LD4BHU	Load four bytes to four half-word registers with zero extension
LD2H	Load two half-words to registers
STB	Store one byte from a register
STH	Store one half-word from a register
STHH	Store one half-word from a register high
STW	Store one word from a register
ST2W	Store two words from registers
ST4HB	Store four bytes from four half-word registers
ST2H	Store two half-words from registers
MODDEC	Decrement a register value by a 5-bit immediate value
MODINC	Increment a register value by a 5-bit immediate value

- Transfer instructions

MVFSYS	Move a control register to a general purpose register
MVTSYS	Move a general purpose register to a control register
MVFACC	Move a word from an accumulator
MVTACC	Move two general purpose registers to an accumulator

- Compare instructions

CMPcc	Compare cc = EQ (000), NE (001), GT (010), GE (011), LT (100), LE (101), PS - both positive (110), NG - both negative (111)
CMPUcc	Compare unsigned cc = GT (010), GE (011), LT (100), LE (101)

【図１７】

- Arithmetic operation instructions

ABS	Absolute
ADD	Add
ADDC	Add with carry
ADDHppp	Add half-word
	ppp = LLL (000), LLH (001), LHL (010), LHH (011), HLL (100), HLH (101), HHL (110), HHH (111)
ADDS	Add register Rb with the sign of the third operand
ADDS2H	Add sign to two half-word
ADD2H	Add two pairs of half-words
AVG	Average with rounding towards positive infinity
AVG2H	Average two pairs of half-words rounding towards positive infinity
JOINpp	Join two half-words
	pp = LL (00), LH (01), HL (10), HH (11)
SUB	Subtract
SUBB	Subtract with borrow
SUBHppp	Subtract half-word
	ppp = LLL (000), LLH (001), LHL (010), LHH (011), HLL (100), HLH (101), HHL (110), HHH (111)
SUB2H	Subtract two pairs of half-words

- Logical operation instructions

AND	logical AND
OR	logical OR
NOT	logical NOT
XOR	logical exclusive OR
ANDFG	logical AND flags
ORFG	logical OR flags
NOTFG	logical NOT a flag
XORFG	logical exclusive OR flags

- Shift operation instructions

SRA	Shift right arithmetic	
SRAHp	Shift right arithmetic a half-word	p = L (0), H (1)
SRA2H	Shift right arithmetic two half-words	
SRC	Shift right concatenated registers	
SRL	Shift right logical	
SRLHp	Shift right logical a half-word	p = L (0), H (1)
SRL2H	Shift right logical two half-words	
ROT	Rotate right	
ROT2H	Rotate right two half-words	

- Bit operation instructions

BCLR	Clear a bit
BNOT	Invert a bit
BSET	Set a bit
BTST	Test a bit

【図１８】

- Branch instructions

BRA	Branch
BRATZR	Branch if zero
BRATNZ	Branch if not zero
BSR	Branch to subroutine
BSRTZR	Branch to subroutine if zero
BSRTNZ	Branch to subroutine if not zero
DBRA	Delayed Branch
DBRAI	Delayed Branch immediate
DBSR	Delayed Branch to subroutine
DBSRI	Delayed Branch immediate to subroutine
DJMP	Delayed Jump
DJMPI	Delayed Jump immediate
DJSR	Delayed Jump to subroutine
DJSRI	Delayed Jump immediate to subroutine
JMP	Jump
JMPTZR	Jump if zero
JMPTNZ	Jump if not zero
JSR	Jump to subroutine
JSRTZR	Jump to subroutine if zero
JSRTNZ	Jump to subroutine if not zero
NOP	No operation

- OS-related instructions

TRAP	Trap
REIT	Return from exception, interrupts, and traps

- DSP Arithmetic operation instructions

MUL	Multiply	
MULX	Multiply with extended precision	
MULXS	Multiply and shift to the left by one with extended precision	
MULX2H	Multiply two pairs of half-words with extended precision	
MULHXpp	Multiply two half-words with extended precision pp = LL (00), LH (01), HL (10), HH (11)	
MUL2H	Multiply two pairs of half-words	
MACd	Multiply and add	(d = 0, 1)
MACSd	Multiply, shift to the left by one, and add	(d = 0, 1)
MSUBd	Multiply and subtract	(d = 0, 1)
MSUBSd	Multiply, shift to the left by one, and subtract	(d = 0, 1)
SAT	Saturate	
SATHH	Saturate word operand into high half-word	
SATHL	Saturate word operand into low half-word	
SATZ	Saturate into positive number	
SATZ2H	Saturate two half-words into positive number	
SAT2H	Saturate two half-word operands	

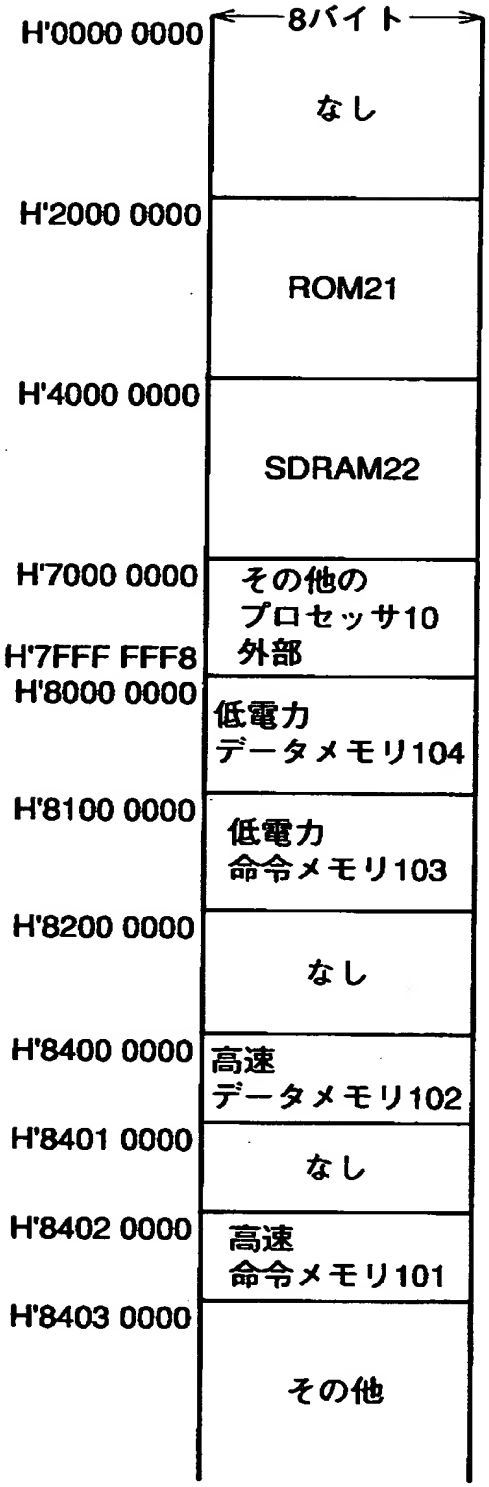
- Repeat instructions

REPEAT0	Repeat a block of instructions #0
REPEAT1	Repeat a block of instructions #1

- Debugger supporting instructions

DBT	Debug trap
RTD	Return from debug interrupt and trap

【図１９】


8バイト
H'0000 0000
なし
H'2000 0000
ROM21
H'4000 0000
SDRAM22
H'7000 0000
その他の
プロセッサ10
外部
H'7FFF FFF8
H'8000 0000
低電力
データメモリ104
H'8100 0000
低電力
命令メモリ103
H'8200 0000
なし
H'8400 0000
高速
データメモリ102
H'8401 0000
なし
H'8402 0000
高速
命令メモリ101
H'8403 0000
その他

【図２０】

●Operation:

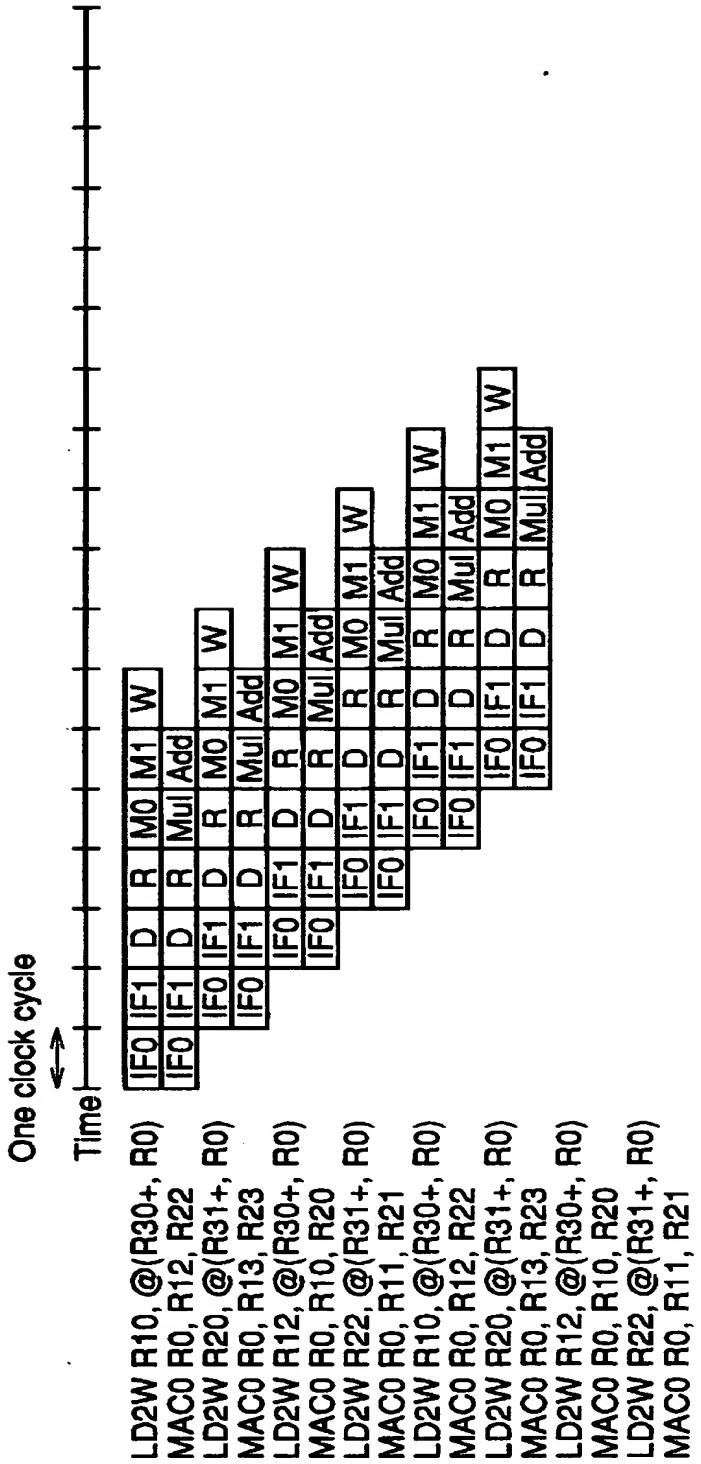
```
REPEAT1 #count, #pcaddr
        RPT1_C = #count-1
        RPT1_S = PC + 8
        RPT1_E = PC + pcaddr
        RPT1_I(0:5) = Instructions at memory((PC+8):(PC+48))
        if (PC ==RPT1_E && RPT1_C > 0) {
                RPT1_C--
                PC == RPT1_S
        }
```

●Example:

```
        REPEAT1 #20, #48
START:LD2W R10, @(R30+, R0) ‖ MAC0 R0, R12, R22
        LD2W R20, @(R31+, R0) ‖ MAC0 R0, R13, R23
        LD2W R12, @(R30+, R0) ‖ MAC0 R0, R14, R24
        LD2W R22, @(R31+, R0) ‖ MAC0 R0, R15, R25
        LD2W R14, @(R30+, R0) ‖ MAC0 R0, R16, R26
END:LD2W R24, @(R31+, R0) ‖ MAC0 R0, R17, R27
```

Zero-overhead loop

【図２１】


One clock cycle
Time
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
IF0 IF1 D R M0 M1 W
IF0 IF1 D R Mul Add
LD2W R10, @(R30+, R0)
MAC0 R0, R12, R22
LD2W R20, @(R31+, R0)
MAC0 R0, R13, R23
LD2W R12, @(R30+, R0)
MAC0 R0, R10, R20
LD2W R22, @(R31+, R0)
MAC0 R0, R11, R21
LD2W R10, @(R30+, R0)
MAC0 R0, R12, R22
LD2W R20, @(R31+, R0)
MAC0 R0, R13, R23
LD2W R12, @(R30+, R0)
MAC0 R0, R10, R20
LD2W R22, @(R31+, R0)
MAC0 R0, R11, R21

【図２２】

●Operation:

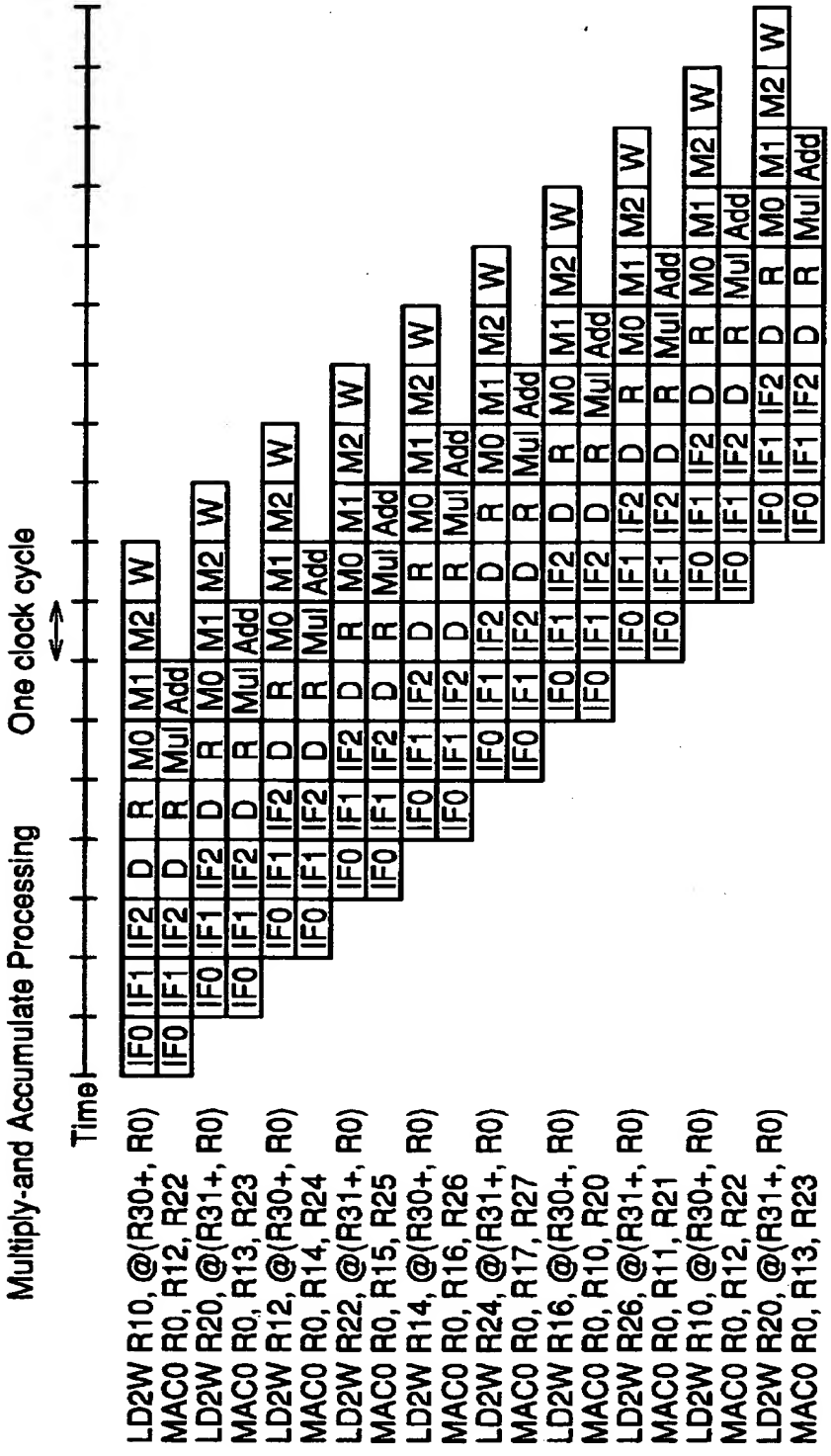
```
REPEAT0 #count, #pcaddr
        RPT0_C = #count-1
        RPT0_S = PC + 8
        RPT0_E = PC + pcaddr
        RPT0_I = Instruction at memory(PC+8)
        if (PC ==RPT0_E && RPT0_C > 0) {
          RPT0_C--
          PC == RPT0_S
        }
```

●Example:

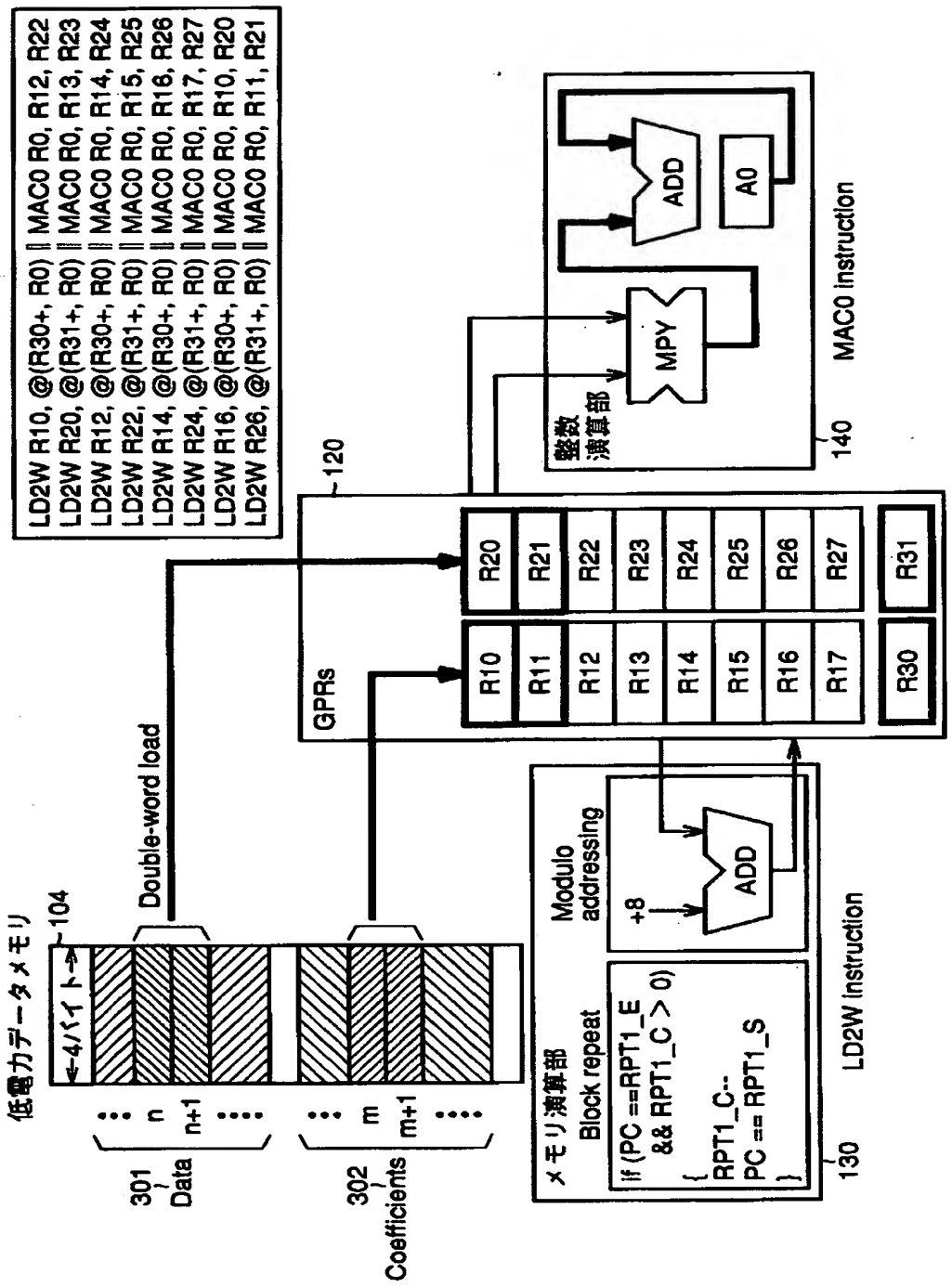
```
        REPEAT0 #10, #64
START:LD2W R10, @(R30+, R0) ‖ MAC0 R0, R12, R22
        LD2W R20, @(R31+, R0) ‖ MAC0 R0, R13, R23
        LD2W R12, @(R30+, R0) ‖ MAC0 R0, R14, R24
        LD2W R22, @(R31+, R0) ‖ MAC0 R0, R15, R25
        LD2W R14, @(R30+, R0) ‖ MAC0 R0, R16, R26
        LD2W R24, @(R31+, R0) ‖ MAC0 R0, R17, R27
        LD2W R16, @(R30+, R0) ‖ MAC0 R0, R10, R20
  END:LD2W R26, @(R31+, R0) ‖ MAC0 R0, R11, R21
```

Zero-overhead loop

【図２３】


Multiply-and Accumulate Processing
One clock cycle
Time
LD2W R10, @(R30+, R0)
MAC0 R0, R12, R22
LD2W R20, @(R31+, R0)
MAC0 R0, R13, R23
LD2W R12, @(R30+, R0)
MAC0 R0, R14, R24
LD2W R22, @(R31+, R0)
MAC0 R0, R15, R25
LD2W R14, @(R30+, R0)
MAC0 R0, R16, R26
LD2W R24, @(R31+, R0)
MAC0 R0, R17, R27
LD2W R16, @(R30+, R0)
MAC0 R0, R10, R20
LD2W R26, @(R31+, R0)
MAC0 R0, R11, R21
LD2W R10, @(R30+, R0)
MAC0 R0, R12, R22
LD2W R20, @(R31+, R0)
MAC0 R0, R13, R23
IF0 IF1 IF2 D R M0 M1 M2 W
Mul Add

【図２４】


LD2W R10, @(R30+, R0) ‖ MAC0 R0, R12, R22
LD2W R20, @(R31+, R0) ‖ MAC0 R0, R13, R23
LD2W R12, @(R30+, R0) ‖ MAC0 R0, R14, R24
LD2W R22, @(R31+, R0) ‖ MAC0 R0, R15, R25
LD2W R14, @(R30+, R0) ‖ MAC0 R0, R16, R26
LD2W R24, @(R31+, R0) ‖ MAC0 R0, R17, R27
LD2W R16, @(R30+, R0) ‖ MAC0 R0, R10, R20
LD2W R26, @(R31+, R0) ‖ MAC0 R0, R11, R21
120
GPRs
R20
R21
R22
R23
R24
R25
R26
R27
R31
R10
R11
R12
R13
R14
R15
R16
R17
R30
整数演算部
MPY
ADD
A0
140
MAC0 instruction
低電力データメモリ
104
4バイト
Double-word load
301
Data
n
n+1
302
Coefficients
m
m+1
メモリ演算部
Modulo addressing
Block repeat
+8
ADD
if (PC ==RPT1_E
&& RPT1_C > 0)
{
RPT1_C--
PC == RPT1_S
}
130
LD2W instruction

【書類名】　　要約書

【要約】

【課題】　低速で低消費電力のメモリを用いつつも高速に処理することができ、処理性能の向上を図ることが可能なデータ処理装置を提供すること。

【解決手段】　複数のメモリバンクから命令をフェッチする場合に、メモリバンクの選択に対応したパイプラインステージＩＦ０と、命令の読出しに対応したパイプラインステージＩＦ１とを発生させてパイプライン処理を行なう。したがって、選択したメモリバンクのみをプリチャージすることができ、消費電力を削減することが可能となる。また、パイプラインステージＩＦ０とＩＦ１とが並列に行なわれるため、命令メモリのスループットを向上させることが可能となる。

【選択図】　　図８

出願人履歴情報

識別番号　〔０００００６０１３〕

１．変更年月日　１９９０年　８月２４日
［変更理由］　新規登録
住　所　東京都千代田区丸の内２丁目２番３号
氏　名　三菱電機株式会社